Titre: Title:	Security mechanisms for next-generation mobile networks
Auteur: Author:	Yao Hui Lei
Date:	2008
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Lei, Y. H. (2008). Security mechanisms for next-generation mobile networks [Thèse de doctorat, École Polytechnique de Montréal]. PolyPublie. https://publications.polymtl.ca/8137/

URL de PolyPublie: PolyPublie URL:	https://publications.polymtl.ca/8137/
Directeurs de recherche: Advisors:	Samuel Pierre, & Alejandro Quintero
Programme: Program:	Génie informatique

UNIVERSITÉ DE MONTRÉAL

SECURITY MECHANISMS FOR NEXT-GENERATION MOBILE NETWORKS

YAO HUI LEI

DÉPARTEMENT DE GÉNIE INFORMATIQUE

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

THÈSE PRÉSENTÉE EN VUE DE L'OBTENTION

DU DIPLÔME DE PHILOSOPHIÆ DOCTOR

(GÉNIE INFORMATIQUE)

MARS 2008

Your file Votre référence
ISBN: 978-0-494-41752-2
Our file Notre référence
ISBN: 978-0-494-41752-2

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Cette thèse intitulée:

SECURITY MECHANISMS FOR NEXT-GENERATION MOBILE NETWORKS

présentée par: LEI Yao Hui

en vue de l'obtention du diplôme de: Philosophiæ Doctor

a été dûment acceptée par le jury d'examen constitué de:

Mme. NICOLESCU Gabriela, Doct., présidente

M. PIERRE Samuel, Ph.D., membre et directeur de recherche

M. QUINTERO Alejandro, Doct., membre et codirecteur de recherche

M. MULLINS John, Ph.D., membre

M. KHENDEK Ferhat, Ph.D., membre

To my family, including my unborn son...

ACKNOWLEDGMENTS

I thank God for giving me the strength, persistence, and determination to complete the thesis. I also have many people to thank, as I have learned so much from so many. I would like to thank all those people who made this thesis possible and an unforgettable experience for me.

First of all I wish to express my sincere gratitude to my director, Samuel Pierre, who guided this thesis and helped whenever I was in need. I think his presence at the LARIM was the best thing that could happen to me and my thesis. I am deeply indebted for his constant support. Without his help, this thesis would not be possible.

I gratefully acknowledge Alejandro Quintero, my co-director, for his continuous guidance, corrections and advices throughout the course of the thesis. Particularly, I appreciate his participation in several subjects discussion.

Finally, I wish to thank my parents and my wife for all their endless love, understanding, support and encouragement.

ABSTRACT

The mobile networks are characterized by limited powerful mobile devices, limited bandwidth and wireless link. Consequently, any transmitted message can be intercepted by an unknown number of other users. These users can also be malicious and brings attacks to the networks. In addition, due to the mobility of mobile users, authentication is critical while the user roaming to another area to access mobile services.

This thesis studies security mechanisms for next-generation mobile networks. The main research objective of this thesis is to design a set of efficient and cost-effective security mechanisms dedicated to next generation mobile networks. Particularly, this thesis addresses three major issues: mobile service access, enhancement of UMTS AKA and DoS attacks in UMTS. This thesis comprises three parts.

In the first part, the thesis gives an introduction and literature review in Chapter 1 and Chapter 2. This part shows the characteristics and constraints in the design of security mechanisms in mobile networks. Then, we point out the motivation and the research challenges. The research objectives of the whole thesis are presented. Last, we analyze current works on these issues.

In the second part, four novel mechanisms are designed to give solutions to three issues while accommodating constraints in mobile networks. Chapter 3 shows a lightweight secure and accountable mobile service access through reusable tickets. This work has several advantages compared to related work. First, the ticket is reusable. Second, the mechanism is lightweight at mobile devices. Last, the charging and billing is practical and may avoid payment chain reinitialization. Chapter 4 presents an enhancement of UMTS AKA through a technique: vector combination. As a result, we eliminate the synchronization of sequence numbers and liberate the HE/HLR from the bottleneck

of authentication traffic generation. Chapter 5 presents another enhancement from the mobility point of view in UMTS. Chapter 6 addresses the DoS issue in UMTS. We design a client puzzle against DoS attacks. The puzzle is based on quasi partial collision in a hash function. Its difficulty levels are fine-grained compared to related approaches where there is a gap between two neighbor difficulty levels.

In the last part, this thesis discusses generally the proposed mechanisms. It also studies their limitations. In the end of this thesis, a conclusion is given.

RÉSUMÉ

Les réseaux mobiles sont caractérisés par les dispositifs mobiles avec capacité limités, la largeur de bande limitée et le lien sans fil. En conséquence, n'importe quel message transmis peut être arrêté par un nombre inconnu d'autres utilisateurs. Ces utilisateurs peuvent également être malveillants et apportent des attaques aux réseaux. En outre, en raison de la mobilité des utilisateurs mobiles, authentification est critique tandis que l'utilisateur errant à un autre secteur pour accéder à des services mobiles.

Cette thèse étudie des mécanismes de sécurité pour les réseaux mobiles de la prochaine génération. L'objectif principal de recherches de cette thèse est de concevoir un ensemble de mécanismes sécurité efficaces consacrés aux réseaux de mobile de prochaine génération. En particulier, cette thèse aborde trois questions principales : accès à service mobile, amélioration d'UMTS AKA et attaques de DoS dans UMTS. Cette thèse contient trois parties.

Dans la première partie, cette thèse donne une introduction et une revue de littérature dans Chapitre 1 et Chapitre 2. Cette partie montre les caractéristiques et les contraintes dans la conception des mécanismes de sécurité dans les réseaux mobiles. Après, nous précisons la motivation et les défis de recherches. Les objectifs de recherches de la thèse entière sont présentés. À la fin, nous analyzons les travaux courants sur ces questions.

Dans la deuxième partie, trois nouveaux mécanismes sont conçus pour donner des solutions à trois issues tout en adaptant à des contraintes dans les réseaux mobiles. Chapitre 3 montre un accès léger et facturable à service mobile par les tickets réutilisables. Ce travail a plusieurs avantages comparés au travail relatif. D'abord, le ticket est réutilisable. Deuxièmement, ce mécanisme est léger dans la côté de terminaux mobiles. Dernièrement, la facturation est pratique et peut éviter la réinitialisation

à chaînes de paiement. Chapitre 4 présente une amélioration d'UMTS AKA par la technique: combinaison de vecteur. En conséquence, nous éliminons la synchronisation des nombres de séquences et libérons le HE/HLR du goulot d'étranglement de la génération du trafic d'authentification. Chapitre 5 présente un autre amélioration du point de vue de mobilité dans UMTS. Chapitre 6 aborde la question de DoS dans UMTS. Nous concevons un puzzle de client contre attaques de DoS dans UMTS. Le puzzle est basé sur la collision partielle quasi dans une fonction de hachage. Ses niveaux de difficulté sont à grain fin comparés aux approches relatives où il y a un espace entre deux niveaux voisins de difficulté.

Dans la dernière partie, cette thèse discute généralement les mécanismes proposés. On étudie également leurs limitations. À la fin de cette thèse, une conclusion est donnée.

CONDENSÉ

Cette thèse étudie des mécanismes de sécurité dans les réseaux mobiles de prochaine génération. Les réseaux mobiles ont leurs caractéristiques particulières. Une caractéristique significative des réseaux mobiles est celui de la communication sans fil. La capacité de communiquer avec d'autres unités mobiles ou stations de base fixes est accomplie par la transmission de radio. Une conséquence de cette communication est que les messages transmis peuvent, en principe, être reçus par un nombre inconnu d'autres utilisateurs. En plus, la communication sans fil est également plus sensible aux attaques actives. Un adversaire peut brancher sur le canal de transmission sans être détecté.

Les réseaux mobiles sont caractérisés par les dispositifs mobiles avec capacité limitée, la largeur de bande limitée et lien sans fil. En conséquence, n'importe quel message transmis peut être arrêté par un nombre inconnu d'autres utilisateurs. Ces utilisateurs peuvent également être malveillants et apporter des attaques aux réseaux. En outre, en raison de la mobilité des utilisateurs mobiles, l'authentification est critique lorsque l'utilisateur entre à un autre domain pour accéder à des services mobiles.

Donc, un ensemble de mécanismes doit être conçu pour ces dispositifs mobiles de sorte que l'authentification mutuelle et l'intimité numérique puissent être accomplie efficacement. En général, elles devraient être légères, exigeant des interventions minimales d'utilisateur, simples à implementer, ayant un niveau de sécurité et scalabilité.

L'objectif principal de cette thèse est de concevoir un ensemble de mécanismes de sécurité efficaces consacrés aux réseaux mobile de prochaine génération. Plus spécifiquement, il vise à atteindre les objectifs spécifiques suivants:

1. *Proposer* un mécanisme d'authentification pour l'accès de service à valeur

ajoutée qui tient compte des contraintes dans les réseaux mobiles;

2. *Proposer* des améliorations de l'authentification et d'accord de clef dans UMTS;
3. *Concevoir* un mécanisme de puzzle de client contre des attaques de DoS dans UMTS;
4. *Valider et évaluer* les mécanismes proposés afin de vérifier leur efficacité et leur robustesse.

Cette thèse aborde trois questions principales: accès de service mobile, améliorations d'UMTS AKA et contre attaques de DoS dans UMTS.

1 Accès de service mobile par ticket réutilisable

1.1 Introduction

Dans les systèmes mobiles de la prochaine génération, le service de télécommunications mobile universel (UMTS) apporte aux utilisateurs mobiles une large gamme de nouveaux services à valeur ajoutée (VASs). Pour utiliser un service, les utilisateurs doivent s'authentifier au fournisseur de service à valeur ajoutée (VASP), alors que la dernière soit aussi authentifiée.

Étant donné les capacités limitées de calcul des dispositifs mobiles, concevoir des mécanismes de sécurité pour accès de service appropriés aux réseaux mobiles est un défi non trivial. En raison de la complexité des problèmes fondamentaux, la cryptographie publique (PKC) ou la cryptographie asymétrique comporte souvent des opérations plus chères que des opérations dans la cryptographie symétrique. Donc, dans la conception de protocole, il est toujours préférable de déplacer des requis de calcul du dispositif mobile à un serveur puissant.

Dans des communications mobiles, un ticket est un ensemble de données qui est analogue à un ticket que nous utilisons dans divers événements sociaux. Ainsi, il n'y a aucun besoin d'exécuter des protocoles de distance longue avec une troisième partie en ligne afin de valider le certificat de l'utilisateur.

Il y a plusieurs avantages à employer des tickets pour l'accès de service mobile:

- *Flexibilité.* Les utilisateurs peuvent facilement établir et mettre à jour des profils personnalisés de service en achetant l'ensemble de tickets appropriés. Ils ne doivent pas s'engager dans des rapports contractuels à long terme avec des fournisseurs de service. Au lieu de cela, ils peuvent choisir des services comme ils en ont besoin.

- *Scalabilité.* Pour que le fournisseur de service décide de fournir le service, le ticket doit contenir toute l'information nécessaire. Ainsi, il n'y a aucun besoin d'exécuter des protocoles avec un agent de confiance pour effectuer l'authentification.

- *Intimité numérique.* Le mécanisme basé sur un ticket, ainsi qu'un agent de confiance qui publie ou acquiert des tickets et manipule des paiements au nom des utilisateurs, peuvent efficacement protéger l'intimité numérique des utilisateurs. L'utilisateur peut seulement démontrer qu'il est un support légitime du ticket, et il ne doit pas nécessairement indiquer son identité vraie. Ainsi, aucune information privée n'est disponible aux fournisseurs de service.

Les utilisateurs ne doivent pas révéler l'information personnelle, quand l'authentification de ticket est exigée. En outre, des billets peuvent être employés dans des VASPs différents, qui sont de grande importance dans les réseaux mobiles hétérogènes.

1.2 Techniques Légères de non-reniement

Le non-reniement est exigé pour l'accès de service. En raison des capacités limitées de calcul des dispositifs mobiles, la technique de non-reniement devrait être légère du côté d'utilisateur mobile. La signature numérique est utilisée souvent pour le non-reniement. Cependant, les calculs sont souvent coûteux. Les fonctions de hachage sont plus rapides que des algorithmes de signature numérique.

Une chaîne de fonction de hachage est une chaîne à laquelle on a appliqué qu'une fonction de hachage plusieurs fois. Ainsi, une chaîne haché peut être employée pour le non-reniement plusieurs fois. Dans ce mécanisme d'accès de service mobile, nous adoptons des chaînes de fonction haché dans nos protocoles conçus pour fournir le non-reniement.

1.3 Accès de service par ticket

Nous définissons d'abord un modèle d'accès de service dans lequel il y a quatre entités: un certificat d'autorité, un serveur de ticket, un utilisateur mobile et le fournisseur de service. Le certificat d'autorité est responsable de délivrer des certificats à l'utilisateur mobile et produit des paires principales de publique / privé au serveur de ticket et au fournisseur de service. Puis, l'utilisateur mobile demande un billet à partir du serveur de ticket. Plus tard, il peut employer ce ticket pour service d'accès. Une fois que l'utilisateur mobile obtient le ticket, il peut l'employer pour demander le service du fournisseur n'importe quand, n'importe où.

Ce mécanisme contient deux protocoles: acquisition d'un ticket et fourniture de service. Dans le protocole d'acquisition d'un ticket, l'utilisateur mobile présente son certificat au serveur de ticket. Quand le serveur de ticket valide le certificat avec succès, il génère un ticket à l'utilisateur mobile. Ce ticket consistue une chaîne hachée.

Ainsi, l'utilisateur mobile peut réutiliser ce ticket jusqu'à ce que la chaîne haché soit épuisée. De plus, ce ticket contient une autre chaîne hachée pour facturation.

Dans le protocole de la fourniture de service, l'utilisateur mobile présente son ticket au fournisseur de service. Quand le fournisseur de service valide ce ticket avec succès, il enlève un noeud de la chaîne et établit une clef secrète pour communiquer avec l'utilisateur mobile. Selon le type de service, le fournisseur peut demander à l'utilisateur mobile de mettre à jour la chaîne de facturation. Bien sûr, la chaîne haché de facturation joue le même rôle : non-reniement pour l'utilisateur et le fournisseur.

1.4 Analyse de sécurité et comparaison

Le but d'un mécanisme d'accès est de s'assurer que l'authentification est réalisée pour tous les entités impliqués dans la communication. Nous analysons la sécurité de notre mécanisme par la logique de BAN. Cette logique peut valider un protocole afin de justifier si deux principaux établissent un rapport de confiance. Par l'analyse de la logique de BAN, nous croyons que nos protocoles réussissent à établir le rapport de confiance entre l'utilisateur mobile, le serveur de ticket et le fournisseur de service.

Nous avons également comparé notre mécanisme à d'autres travaux. Vu les capacités limitées de calcul des dispositifs mobiles, le mécanisme est léger du côté de l'utilisateur mobile : seulement les opérations symétriques et les fonctions de hachage sont nécessaires. Nous transférons la complexité informatique élevée au serveur de ticket et au fournisseur de service qui sont habituellement plus puissants. Deux chaînes hachés sont présentées pour réaliser la réutilisation de ticket, mutuelle d'authentification et non-reniement. En permettant la réutilisation de ticket, un utilisateur mobile ne doive pas appliquer un nouveau ticket pour chaque accès de service. C'est trè interssant pour deux raisons: d'abord, il est approprié pour les utilisateurs mobiles; puis,

il diminue la charge du serveur de ticket.

2 Améliorations d'UMTS AKA

2.1 Introduction

Un mécanisme important dans UMTS est l'authentification et l'accord de clef (AKA). UMTS AKA réalise l'authentification mutuelle pour l'utilisateur et le réseau par une clef secrète partagée à long terme entre le module universel d'identité de l'abonné de l'utilisateur (USIM) et le centre d'authentification (AuC).

L'USIM de l'utilisateur mobile maintient un nombre de séquence avec le réseau de maison de l'utilisateur. Quand l'utilisateur mobile entre dans un réseau visité, le réseau de maison envoie un ensemble de vecteurs d'authentification au réseau visité. Pour authentifier la station mobile, le réseau visité prend un vecteur inutilisé et envoie la pièce de défi à la station mobile. La station mobile vérifie la fraîcheur du nombre de séquence et calcule une réponse. En outre, la station mobile calcule la clef de chiffre et la clef d'intégrité pour la communication suivante. Le réseau visité compare la réponse à la réponse prévue dans le vecteur d'authentification. Si elles sont les mêmes, la station mobile est authentifiée. Alors, l'authentification mutuelle est réalisée dans UMTS.

2.2 UMTS AKA faiblesses

Cependant, UMTS AKA a été critiqué pour son introduction du nombre de séquence. Un échec de synchronisation de nombre de séquence pourrait amène l'exécution compliquée de protocole particulièrement quand l'utilisateur erre à un réseau étranger.

Un autre inconvénient d'UMTS AKA réside dans la distribution des vecteurs

d'authentification. Un choix de la taille n de vecteurs d'authentification peut être employé pour l'authentification de n fois seulement. Puisque le réseau de maison est responsable de générer et d'envoyer des vecteurs d'authentification pour tous les abonnés.

2.3 Amélioration d'UMTS AKA: VC-AKA

Nous abandonnons l'adoption des nombres de séquence et proposons un AKA amélioré pour éliminer les inconvénients ci-dessus. L'AKA proposé est basé sur la combinaison de vecteur : différents vecteurs combinés ensemble peuvent également être employés pour l'authentification. Par la combinaison de vecteur, un choix de la taille n de vecteurs d'authentification peut réaliser jusqu'à l'authentification mutuelle de 2^{n-1} fois au lieu seulement de n fois dans UMTS AKA. Alors, le trafic du réseau de maison pour générer des vecteurs d'authentification est exponentiellement diminué.

Ce mécanisme consiste en deux procédures. Dans la première procédure, le réseau de la maison génère une séquence de vecteurs et l'envoie au réseau visité. En outre, l'utilisateur mobile génère une autre séquence de vecteurs par lui-même. Les deux vecteurs ne sont pas identiques. C'est critique pour empêcher des attaques par rejoue de message.

Dans la deuxième procédure, le réseau visité envoie une combinaison des vecteurs comme un défi à l'utilisateur mobile. L'utilisateur mobile vérifie d'abord si cette combinaison a été utilisée. Si c'est le cas, il rejette l'authentification. Autrement, il obtient le défi et calcule la réponse. Il met à jour ses combinaisons utilisées et renvoie la réponse au réseau visité. Le réseau visité vérifie le défi. S'il réussit, l'authentification mutuelle entre le réseau visité et l'utilisateur mobile est réalisée.

2.4 Amélioration d'UMTS AKA: MO-AKA

Dans notre vrai vie, bien que personne n'ait le potentiel de comportement aléatoire, il existe des routines dans la vie de chaque personne. Ainsi, dans un réseau mobile, beaucoup d'utilisateur suivent régulierement une routine chaque jour.

Nous considérons l'authentification d'utilisateur dans UMTS du point de vue des modèles de mobilité d'utilisateur. L'UMTS AKA ne considère pas le modèle de mobilité d'utilisateur. Il traite la mobilité pour tous les utilisateurs mobiles d'une manière générale : la même authentification par un mécanisme de défi - réponse. Le mécanisme de défi - réponse implique toujours trois entités : l'utilisateur mobile, le réseau de la maison (HE/HLR) et le réseau visité (VLR). En conséquence, ils ne sont pas efficaces si les utilisateurs mobiles restent dans leurs réseaux étrangers fréquemment visités pour la plupart de leurs activités.

Pour améliorer le mécanisme d'AKA à mobilité orientée, nous proposons le MO-AKA dans lequel chaque utilisateur mobile maintient une cachette du VLRs le plus fréquemment visité et du dernier VLR visité. Chaque VLR dans la cachette peut être un proxy de l'HE/HLR pour le but d'authentification. Sur la prochaine authentification, l'utilisateur mobiles cherche d'abord la cachette pour voir s'il y a un rapport secret approprié disponible. S'il peut en trouver un, l'utilisateur mobile et les VLR visités n'ont pas besoin de démarrer le protocole long avec le HE/HLR pour l'authentification.

Par l'analyse des valeurs et la comparaison d'exécution, sur une trace d'une semaine, dans la pratique, nous prouvons que notre approche est plus efficace et sûre.

3 Contre attaques de DoS

3.1 Introduction

La convergence de la communication mobile et de l'Internet apporte des services de l'Internet aux utilisateurs mobiles. L'architecture et les protocoles de système d'UMTS sont conçus pour s'adapter à ces services, alors qu'elles aussi exposent le réseau d'UMTS à des attaques diverses. L'attaque du déni-de-service (DoS) est l'une de ces attaques. Le réseau mobile de terre publique d'UMTS (PLMN) peut être facilement encombré et donc paralysé par le faux trafic de l'Internet.

En outre, les dispositifs mobiles de plus en plus puissants peuvent se coordonner pour lancer des attaques de DoS, tirant profit des ressources limitées dans le réseau, tel que la largeur de bande de lien par radio.

La fonction d'une attaque de DoS est fondamentalement d'inonder ses serveurs de cible avec de nombreux connections des demandes et de les empêcher d'être accessibles à toutes les autres demandes d'utilisateurs légitimes ou de fournir des services. En conséquence, les serveurs de cible deviennent trop surchargés pour répondre aux demandes de leur attaquants. De plus, ils n'ont plus de ressources de système suffisantes pour répondre aux demandes légitimes sur le réseau.

3.2 Puzzle de client

Le puzzle de client est une méthode pour défendre des attaques de DoS. Pour chaque demande de service, le client est forcé de résoudre un puzzle cryptographique avant que le serveur n'affecte ses ressources. Les efforts de calcul pour chaque demande sont très faciles. Cependant, ceci impose une grandd charge sur des adversaires produisant du trafic en grande quantité. L'idée principale derrière des puzzles de client est que

n'importe quel client demandant le service doit allouer certaines de ses ressources propres (durée de la transformation ou mémoire) avant que le serveur n'affecte ses ressources pour la communication suivante. Le client doit consommer des ressources pour résoudre le puzzle, alors que les puzzles sont souvent vérifiés facilement par les serveurs.

Aura a montré un puzzle de client basé sur la collision partielle dans une fonction de hachage. Une bonne fonction de hachage (comme SHA1, MD5) exige qu'aucune des deux entrées distinctes dont les résultats de hachage soient mêmes. Une collision partielle résulte du fait que deux valeurs hachées ont les valeurs partiellement identiques.

Ces puzzles de client, cependant, sont souvent conçus pour l'environnement de réseau fixe. La plupart d'entre eux ne sont pas assez petits pour s'adapter aux réseaux d'accès sans fil. En outre, les difficultés de ces puzzles ne doivent pas facilement être ajustées finement, ce qui a comme conséquence un espace entre deux niveaux voisins de difficulté.

3.3 Puzzle basé sur la fonction de hachage avec collision quasi partielle

Nous avons conçu un puzzle de client basé sur la fonction de hachage avec collision quasi partielle. Une collision dans une fonction de hachage est définie comme suit. Nous divisons la partie de collision dans le puzzle partiel régulier de collision en deux parties. La première partie est tous bits 0; la deuxième partie peut être d'autres valeurs prédéfinies y compris tous bits 0. Ainsi, la deuxième partie ont plus de choix au lieu seulement un choix (tous bits 0) dans le puzzle régulier.

De cette façon, nous pouvons ajouter plus de difficultés entre deux niveaux de difficulté dans des puzzles avec collision partielle.

Nous avons simulé nos puzzles et avons comparé à d'autres puzzles. Les résultats ont prouvé que nous avons ajouté plus de niveaux de difficultés dans l'espace vide entre deux niveaux voisins dans des puzzles réguliers avec collision partielles.

4 Relation des mécanismes

Dans la communication mobile, l'utilisateur mobile communique avec le serveur par l'intermédiaire du lien sans fil. Le lien sans fil apporte un nouveau défi pour concevoir des mécanismes de sécurité. Quelle couche devrions-nous protéger? La couche lien, la couche réseau ou la couche plus élevée (transport ou couche application)? Ici, nous discutons nos mécanismes dans le cadre de réseau.

La sécurité de la couche lien protège un réseau sans fil en niant l'accès au réseau lui-même avant qu'un utilisateur ne soit pas authentifié avec succès. Ceci empêche des attaques contre l'infrastructure de réseau et protège le réseau contre les attaques qui connectent le réseau par la couche d'IP. Cependant, les protocoles de sécurité de la couche lien ne fournissent pas la sécurité bout à bout. D'autre part, la couche réseau ou la couche transport peut fournir une solution plus appropriée de sécurité dans la partie fixe des réseaux. Ainsi, on accepte généralement que la sécurité de la couche lien puisse être complétée par l'utilisation des mécanismes de sécurité à des couches plus élevées de protocole. Des protocoles de sécurité de couche de lien peuvent être employés pour fournir la sécurité de domaine d'accès de réseau. Les protocoles de la couche élevée peuvent fournir la sécurité bout à bout.

Dans cette thèse, les mécanismes conçus sont particulièrement applicables aux réseaux mobiles de prochaine génération où un certain nombre de systèmes mobiles coexistent à grande échelle. Pour les mettre dans le modèle de réseau, les perfectionnements d'UMTS AKA authentifient les utilisateurs mobiles à la couche lien tandis que l'accès de service par ticket est responsable de fournir le service dans la couche

réseau, couche transport ou même dans la couche application. En conséquence, la disposition de service s'effectura comme suit. L'utilisateur mobile doit d'abord s'authentifier aux réseaux mobiles par les mécanismes d'authentification de la couche lien d'UMTS ; puis, il peut envoyer une demande de disposition de service à son fournisseur de service local par le mécanisme d'accès de service par ticket. Ceci comporterait l'authentification à grande échelle. Le mécanisme peut fournir la flexibilité et la scalabilité dans ce scénario.

Sans aucun doute, les fournisseurs de service et les réseaux d'UMTS pourraient être les cibles potentielles des attaques de DOS pendant la disposition de service. Les puzzles de client conçus peuvent être lancés pour le défendre contre ces attaques si le système le trouve nécessaire. En bref, nos mécanismes conçus fournissent les solutions appropriées d'authentification pour le réseau mobile de deuxième génération non seulement de la couche lien mais également de la couche réseau ou même de la couche élevée. Nous donnons également une solution à grain fin pour des attaques de DOS de potentiel.

5 Conformité de la contribution aux objectifs de recherches

Dans cette thèse, le premier objectif spécifique de recherche était de proposer un mécanisme d'authentification pour l'accès de service à valeur ajoutée qui tienne compte des contraintes dans les réseaux mobiles. L'accès de service mobile par ticket réutilisable est léger pour le dispositif mobile et mutuellement authentifiable entre le dispositif mobile et le fournisseur de service. Il exige les dispositifs mobiles seulement des opérations légères: fonction de hachage et opérations cryptographiques symétriques. Des opérations de calcul chères telles que des opérations cryptographiques publiques sont seulement utilisées dans le serveur de ticket et les fournisseurs de service puissants.

Vu les contraintes dans les réseaux mobiles, ce mécanisme peut empêcher des attaques par rejoue et garantissent la confidentialité, l'intégrité et l'intimité de l'utilisateur. Car le ticket proposé est réutilisable et le paiement est basé sur une chaîne haché par accroissement de valeur d'escalier, il est plus efficace que d'autres mécanismes d'accès de services mobiles. Ce mécanisme satisfait les contraintes et est approprié aux réseaux mobiles. Ainsi, le mécanisme conçu a réalisé l'objectif spécifique proposé de recherche.

Le deuxième objectif spécifique de recherche était de proposer des améliorations de l'authentification et de l'accord de clef dans UMTS. Nous avons proposé deux améliorations dans cette thèse. L'UMTS AKA original a été conçu spécifiquement pour les réseaux mobiles avec la considération des contraintes. Nos améliorations sont basés sur ce mécanisme sans introduire des opérations chères supplémentaires. Il n'y a aucun calcul cher ajouté dans les deux mécanismes, ainsi ils satisfont également les contraintes et sont appropriés à UMTS. Les deux améliorations augmentent l'efficacité de réseau en libérant le HE/HLR du trafic d'authentification. Pour atteindre ce but, une amélioration est basé sur la combinaison de vecteur tandis que l'autre est basé sur le modèle de mobilité d'utilisateur. En outre, tous les deux permettent une augementation de la la sécurité comparée à l'UMTS AKA original. Ainsi, nous avons réalisé l'objectif spécifique proposé de recherche.

Le troisieme objectif spécifique de recherche était de concevoir un mécanisme de puzzle de client contre des attaques de DoS dans UMTS. Le mécanisme que nous avons proposé est basé sur des collisions quasi partielles dans une fonction de hachage. Il est compact et approprié aux dispositifs mobiles. Vu les caractéristiques des réseaux d'UMTS PLMN, le puzzle conçu de client est à grain fin dans la commande de difficulté et efficace pour défendre des attaques de DOS. Ainsi, ce mécanisme a également réalisé l'objectif proposé de recherche.

Le dernier objectif spécifique de recherche était de valider et évaluer les mécanismes

proposés afin de vérifier leur efficacité et leur robustesse. Dans ce chapitre, nous avons déjà donné une analyse de tous nos mécanismes conçus, des aspects des techniques, des attaques, de la sécurité et de l'exécution. Par l'analyse, nous pouvons voir que nos mécanismes conçus sont efficaces, sûrs et robustes.

L'objectif principal de recherche était de concevoir un ensemble de mécanismes de sécurité efficaces consacrés aux réseaux mobiles de prochaine génération. Comme nous l'avons montré, tous les mécanismes conçus utilisent seulement des techniques légères sur les dispositifs mobiles. La discussion de performance a également montré qu'ils étaient efficaces et pourraient fournir la sécurité forte. Tous ces mécanismes sont consacrés aux réseaux mobiles de prochaine génération. Puisque nous avons réalisé tous les objectifs spécifiques de recherche, nous avons réalisé naturellement l'objectif principal de recherche de cette thèse : concevoir un ensemble de mécanismes de sécurité efficaces consacrés aux réseaux mobiles de prochaine génération.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS AND SYMBOLS

Abbreviations

2G	Second Generation Mobile System
3G	Third Generation Mobile System
AK	Anonymity Key
AKA	Authentication and key agreement
AMF	Authentication management field
AS	Authentication Server
AuC	Authentication Center
AUTN	Authentication Token
AV	Authentication Vector
CA	Certification Authority
CK	Cipher Key
DoS	Denial-of-Service
GSM	Global System for Mobile Communications
HE	Home Environment
HLR	Home Location Register
IK	Integrity Key
IMSI	International Mobile Subscriber Identity
KDC	Key Distribution Center
MAC	The message authentication code included in AUTN
MS	Mobile Station

PKC	Public Key Cryptography
PLMN	Public Land Mobile Network
RAND	Random challenge
SQN	Sequence number
SQNHE	Individual sequence number maintained in the HLR/AuC
SQNMS	The highest sequence number the USIM has accepted
SGSN	Serving GPRS Support Node
SIM	(GSM) Subscriber Identity Module
TGS	Ticket Granting Server
TMSI	Temporary Mobile Subscriber Identity
TTP	Third Trusted Party
UMTS	Universal Mobile Telecommunications System
USIM	Universal Subscriber Identity Module
VAS	Value-Added Service
VASP	Value-Added Service Provider
VLR	Visitor Location Register
XRES	Expected Response

Symbols

$\|\|$	Concatenation
$\oplus$	Exclusive or
$f1_K$	Message authentication function used to compute MAC with key K

$f2_K$	Message authentication function used to compute RES and XRES with key K
$f3_K$	Key generating function used to compute CK with key K
$f4_K$	Key generating function used to compute IK with key K
$f5_K$	Key generating function used to compute AK in normal procedures with key K
$f^{(N)}(x)$	A hash chain on seed x with length of N
$E_S(X)$	Public key encryption of message X with S's public key
$\{X\}_K$	Symmetric encryption of message X with a secret key K
$Sig_S(X)$	A digital signature of message X with S's secret signing algorithm

CHAPTER 1

INTRODUCTION

A significant feature of mobile networks is that of wireless communication. The ability to communicate with other mobile units or wired base stations is accomplished via radio broadcasts transmission. One consequence of this communication is that the transmitted messages can, in principle, be received by an unknown number of other users. Even worse, the wireless communication medium is also more difficult to be protected against active attacks and eavesdropping. An eavesdropper can tap into a communication channel over-the-air without being detected. Besides, an adversary can also jam a channel easily. Hence, security is a major concern.

In mobile networks, mobile devices are generally characterized by the limited memory capacity and limited computational power. In addition, the wireless environment is limited in bandwidth and is subject to erratic changes such as weather, terrain and external interferences. These constraints have prevented a simple migration of cryptographic protocols that have been widely adopted in wire-line networks to wireless networks for authentication and security.

Furthermore, due to the mobility of wireless devices, authentication of both wireless devices and base stations becomes paramount important. In wireless environments where a (roaming) mobile user purchases value-added services (VAS) from a value-added service provider (VASP), authentication protocols should mutually authenticate a mobile user and VASP and initialize an appropriate payment mechanism. Encryption provides the means to regain control of both privacy and authentication.

Therefore, a set of mechanisms need to be designed for these low-power wireless devices so that mutual authentication and data privacy can be accomplished efficiently

and effectively in terms of computational complexity, memory demand and bandwidth requirement. In general, they should be light-weighted, requiring minimal user interventions, simple to implement and fast but also maintaining an expected level of security and a certain amount of scalability.

1.1 Basic Concepts and Definitions

Security is considered an important issue for mobile communication networks. In particular, the design of authentication and privacy mechanisms has received considerable research interest. In this section, we give basic concepts and definitions on this issue.

Definition 1 Access network: *In mobile communications, the access network is the portion of a wireless network that connects access nodes to individual subscribers. More simply, it is the last link in a network between the customer premises and the first point of connection to the network infrastructure.*

Definition 2 Authentication: *The act of verifying a claimed identity, in the form of a pre-existing label from a mutually known name space, as the originator of a message (message authentication) or as the end-point of a channel (entity authentication).*

Definition 3 Mutual authentication: *The act of authentication for both originator and destinator of a message.*

Definition 4 Confidentiality *is the assurance that the person receiving is the intended recipient.*

Definition 5 Non-repudiation *is the assurance that the person sending cannot deny participation.*

Non-repudiation is similar to authentication in that it is an asymmetric security service. A simple way to describe the difference between authentication and non-repudiation is that with authentication the recipient himself is confident about the origin of a message but would not necessarily be able to convince anybody else about it, whereas for non-repudiation the recipient is also able to convince third parties. Digital signature is the mechanism used for non-repudiation. From the view of cryptography, a message's authentication code and non-repudiation code can be identical, and the difference between the two services might only depend on the key distribution.

Definition 6 *A* message authentication code (MAC) *is an authentication tag (also called a checksum) derived by applying an authentication scheme, together with a secret key, to a message. Unlike digital signatures, MACs are computed and verified with the same key, so that they can only be verified by the intended recipient.*

Definition 7 Public key cryptography *is an encryption method that uses a two-part key: a public key and a private key. To send an encrypted message to someone, you use the recipient's public key, which can be sent to you via regular e-mail or made available on any public Web site or venue. To decrypt the message, the recipient uses the private key, which he or she keeps secret. Contrast with "secret key cryptography," which uses the same key to encrypt and decrypt.*

Definition 8 Privacy *is a combination of maintaining the confidentiality of information and restriction of the use of the information, as authorized by the information owner.*

Definition 9 Denial of Service (DoS) *A security intrusion which causes a system to be damaged, and where the damage is sufficient to disable at least one of the services offered by that system, is called a Denial of Service.*

1.2 Motivation and Research Challenges

The unprecedented growth of world-wide mobile wireless markets, coupled with advances in communications technology and the accelerated development of services taking place in fixed networks, instigates the emergence of third Generation Mobile Communications System (3G). The Universal Mobile Telecommunications System (UMTS) represents an evolution in terms of capacity, data speeds and new service capabilities from second generation mobile networks.

UMTS is bringing mobile users with a broad range of new value-added services (VASs). To access a service, users must authenticate themselves to the value-added service provider (VASP). For mobile access across multiple service domains, the traditional access mechanisms require the exchange of authentication information between home domain and foreign domain using roaming agreements. This will be much complicated in large scale networks and limits the future mobile applications.

UMTS also provides more secure wireless access security mechanisms. The access security features in UMTS are a superset of those provided in GSM. UMTS provides mutual authentication between the UMTS subscriber, represented by a smart card application known as the USIM, and the network. UMTS security builds on the success of GSM by retaining the security features that have proved to be needed and that are robust. However, UMTS AKA has been criticized due to its introduction of sequence numbers and its vulnerabilities of redirection attacks and active attacks in corrupted networks. Another drawback of UMTS AKA exists in the distribution of authentication vectors. A size n array of authentication vectors can be used for n times authentication. Since only the HE is responsible for generating and sending authentication vectors for all subscribers, it actually becomes the traffic bottleneck.

In UMTS networks, the danger of DoS attacks is threatening different targets [1]:

the communication infrastructure, specific services and the mobile devices. Given the high risk of DoS attacks for UMTS networks, it is vital to provide a systematic countermeasure to thwart these attacks. In addition, the system needs a mechanism to control the access of legitimate users according to current resource consummation.

1.3 Research Objectives

The main research objective of this thesis is to design a set of lightweight and efficient security mechanisms dedicated to next generation mobile networks. More specifically, it aims at reaching the following objectives:

1. *Propose* an authentication mechanism for mobile value-added service access which takes into account the constraints in mobile networks and shifts high computational efforts from limited power mobile devices to powerful servers;
2. *Propose* enhancements of UMTS authentication and key agreement mechanism;
3. *Design* a client puzzle mechanism against DoS attacks in UMTS;
4. *Validate and evaluate* the performance of the proposed mechanisms in order to check their efficiency and robustness.

1.4 Thesis Outline and Contributions

This thesis is organized as follows. Chapter 2 introduces fundamental concepts and techniques to design mobile service access mechanisms, to improve UMTS AKA, and to defend against DoS attacks. Then, major contributions and originalities are presented in Chapter 3 to Chapter 6.

Chapter 3 propose a novel mechanism to access mobile VASs through reusable tickets. Considering the limited computation capability of mobile devices, the mechanism is lightweight on the mobile user side. We transfer high computational complexity to the ticket server and the VASP which are usually more powerful. Two hash chains are introduced to achieve mutual authentication, non-repudiation and ticket reuse.

Chapter 4 proposes a novel enhancement of UMTS AKA based on authentication vector combination to eliminate the weaknesses and enhance the security. We abandon completely the adoption sequence numbers. The main advantage of VC-AKA is that it liberates the HE/HLR from the bottleneck of authentication vectors generation. The security analysis shows that VC-AKA can defend against redirection attacks and active attacks in corrupted networks.

Chapter 5 proposes another enhancement of UMTS AKA based on one observation: a single mobile user usually has a fairly regular routine and could have a *home VLR* in which occur his most activities. Such mobile users have a high proportion of all mobile users. As a result, the home VLR can establish a trust relationship with the user's home network and authenticate the mobile user directly. Thus, we also liberate the HE/HLR from the traffic bottleneck.

Chapter 6 proposes a fine-grained client puzzle based on quasi partial collisions to DoS attacks in UMTS. By introducing quasi partial collisions, we add more difficulty levels between the gap in partial collisions.

After presenting these contributions, a general discussion is given in Chapter 7. At last, we conclude the thesis in Chapter 8.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

In previous chapter, we proposed our research objectives in wireless mobile networks. The development of new and lightweight security mechanisms is a challenging research issue. As adversaries or attackers become familiar with the technology and develop sophisticated ways to compromise security, the risks of wireless technology with resource limited devices are well identified. Many researchers contribute their efforts to propose appropriate mechanisms to decrease the risks.

Generally speaking, Security is a proactive method as we reinforce the network with security even if there is no attack. Consequently, this fact deteriorates the overall system performance. Thus, it is not recommended to use strong cryptographic algorithms such as public key cryptography for authentication on mobile devices with limited capabilities. Given these constraints, it is recommended to develop lightweight authentication protocols that are able to detect an attack, and take countermeasures based on a security policy.

In this chapter, we are going to review current contributions related to our proposed research objectives. First, we will analyze the mechanisms for mobile value-added service access. Mobile service access involves several subjects: authentication, payment and billing. All these processes require security. Second, we are going analyze UMTS security mechanisms. Currently, UMTS security has been widely scrutinized by researchers. Some weaknesses and improvements have been carefully discussed. Finally, we will analyze the DoS attack, one issue that we cannot ignore when apply

security in our networks. We will focus on the client puzzles mechanisms to defend against DoS attacks.

2.2 Mobile Value-Added Service Access

2.2.1 Background

With the increasing use of mobile communication systems in a variety of mobile value-added services (VAS), the provision of security guarantees to service providers and users is becoming an important issue. This is much more important in next-generation mobile networks where coexist numerous service providers and telecommunication operators. Also, the traditional trust model between users and operators is no longer appropriate because certain service providers could be malicious and deliberately leak a legitimate user's authentication information to others in order to help them pass authentication. Even worse, a service provider can concoct a billing and cheat a user's home network in order to gain profit.

Using tickets to access various services is a widely accepted manner in every day life. We use tickets, for instance, to travel on a train or bus, to enter a cinema or museum, etc. Intuitively, a ticket is a piece of information that represents the rights of a certain user to access a certain service provided by a certain service provider. A flight ticket, for instance, can be used by its legitimate holder (user) to fly from one place to another (service) with the air-plane of an air company (service provider). A ticket may also have associated with it a number of conditions (e.g., validity period).

There is nothing to prevent us to use ticket concept in the context of communication systems. The physical realization of a ticket, however, is different, it is realized in a pure electronic form. In mobile communications, ticket is a piece of data that shows that the user is entitled to certain services such as value-added services. The ticket can

contain all the necessary information for the service provider to decide if it should provide the service or not. Thus, there is no need to run long distance protocols with the on-line agent of the accessing user in order to check the validity of the certificate. When authentication is required with tickets, users do not have to disclose personal information. In addition, tickets can be used in different value-added service providers, which is of great importance in heterogeneous mobile networks. This makes ticket based service access an appealing approach in mobile communications where users roam and contact service providers in foreign domains.

Compared to the ticket in our every day life, the mobile ticket has the following differences. First, it is flexible to provide any combinations of services needed in a single mobile ticket. While in the ticket of every day life, it is usually fixed and not easy do so. Second, the ticket of every day life usually comprises money. As a result, if the ticket is lost, the user loses his money. While in mobile ticket, a ticket is just a way to authenticate the user and may not be required to contain the money. Third, a mobile ticket usually has stronger security than our every day life ticket. It is difficult to make a fake mobile ticket. Last, the acquisition of a mobile ticket is through the mobile network and the user does not have to go somewhere to buy it. It is much more convenient that our every day life ticket.

There are several advantages of using tickets for mobile service access:

- *Flexibility.* Users can easily build and update personalized service profiles by buying the appropriate set of tickets. They do not have to engage into long term contractual relationships with service providers. Instead, they can choose services as they need them as best matches their personal requirements.

- *Scalability.* For the service provider to decide if it should provide the service or not, the ticket can contain all the necessary information. Thus, there is no need to run long-distance protocols with some trusted agent of the accessing user in

order to perform authentication.

- *Privacy.* A ticket based mechanism, together with a trusted agent that issues or acquires tickets and handles payments on behalf of users, can efficiently protect the privacy of users. The user only has to demonstrate that he is a legitimate holder of the ticket, and he does not necessarily have to reveal his true identity. Thus, no private information is available to service providers.

2.2.2 Related Works

Usually, the ticket based authentication scheme consists of *the user*, *the value-added service provider*, and the *ticket server*. The ticketing mechanism is split into two phases: *ticket acquisition* and *ticket use*. The user obtains tickets by running the ticket acquisition protocol with the ticket server. The ticket acquisition phase covers activities such as service discovery and selection, as well as possible authentication of both entities and the transfer of payment. The ticket use phase considers the steps required when the user wishes to use the particular service. These tickets can be used to access services anonymously. In the ticket use protocol, the user presents an appropriate ticket to the service provider, which can verify the validity of the ticket and the legitimacy of the user to use it.

Based on this scheme, a secure billing for mobile information services conducted by Advanced Security for Personal Communications Technologies (ASPeCT) in UMTS is described in [2]. This work relies on a third trusted party (TTP) to support mobile telecommunications security services which use public key cryptography. To support the secure billing services, mobile users and VASPs will each register with TTPs acting as Certification Authorities (CAs), which will certify and manage public keys for them. A non-standard, compact public key certificate format has been chosen in ASPeCT because both storage space on a smart card and bandwidth on the air

interface are strictly limited.

When requiring services from the VASP, the user pays for the service by sending micropayment tokens ("ticks") as shown in [3]. The "tick" is based on a hash chain $f^T(\alpha)$. In order to get payment for the i-th tick, the following takes place:

1. The payer sends $\alpha_i = f^{T-i}(\alpha)$ to the recipient.
2. The recipient verifies that $f(\alpha_i) = \alpha_{i-1}$ (α_{i-1} was received in the previous round).

The VASP is able to check the validity of these ticks with a user's credential certificate issued by the UMTS service provider, or an appropriate TTP acting on behalf of the UMTS service provider. This certificate contains the public key for the user. The UMTS service provider will collect the billing information from VASPs and in turn forward to the user. A two phase charging protocol is illustrated. In the first phase, the user and the VASP authenticate each other and agree on a Diffie-Hellman session key. The second phase is responsible for data transfer through user's ticks.

In [4], a ticket-based authentication and payment protocol for mobile communications was presented. Instead of contacting an on-line TTP, the service provider has to verify only the user's ticket and its legitimacy of the user to use it. This protocol includes ticket acquisition and ticket access. The ticket structure contains a serial number sn, a freshly generated public Diffie-Hellamn agreement key g^u, a freshly generated public signature public key P_U.

$$Ticket = \{id_T, sn, g^u, data, TS_T, Sig_T(h(id_T, sn, g^u, data, TS_T))\}$$

Before staring the ticket-acquisition protocol, the user U computes a random number u and the public key agreement key g^u. He also generates the private signing key and

the corresponding public signature verification key P_U.

$$
\begin{aligned}
U \rightarrow T: \quad & g^w \| g^u \| P_U \| \{id_U\}_L \\
T \rightarrow U: \quad & r \| h(L \| r \| id_T \| Ticket) \| Ticket \\
U \rightarrow T: \quad & \{Sig_U(h(g^w \| g^t \| r \| id_T \| Ticket))\}_L
\end{aligned}
$$

In the protocol, U generates a random number w and computes a temporary public key agreement key g^w. U also generates an encryption session key $L = g^{tw}$ using the T's public key agreement key g^t. Then, U sends T the value g^w, g^u and P_U together with his own identity id_u encrypted under the session key L. When receiving the message, T generates a random number r, computes g^{tw} and then a session key L. Then, it sends back the message to U. U checks the message's correctness by computing the hash value $h(L \| r \| id_v)$.

Once the U has the ticket in hand, he can show it to V, a service provider. Here, V needs a public key agreement key certificate $CertV$ assigned by T.

$$
\begin{aligned}
U \rightarrow V: \quad & Ticket \\
V \rightarrow U: \quad & r \| h(K \| r \| id_v) \| TS_v \| CertV \| ch_data \\
U \rightarrow V: \quad & \{Sig_U(h(Ticket \| K \| r \| id_v \| ch_data \| TS_v))
\end{aligned}
$$

Here, V generates a r and computes $K = h(g^u v \| r)$. He demonstrates knowledge of the session key K. Once U receives the message, he checks the hash value's correctness. Then, he signs the hash value of the concatenation data. Thus, V first verifies the signature and acquires charging data ch_data.

In [5,6], another ticket based mobile service access mechanism was proposed. Instead of using Diffie-Hellamn, it uses RSA public key cryptography. The service architecture involves a Trusted Credential Center (TCC), a Trusted Authentication and

Registration Center (TARC) (via UDDI) and a secure ticket based mechanism for service access as shown in Figure 2.1.

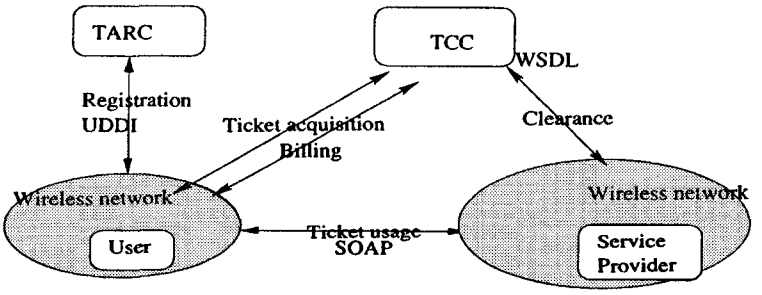

FIGURE 2.1 MOBILE SERVICE MODEL

Users and service providers register with the TARC and are authenticated. Services are described in the TCC and service provider by WSDL. Based on authentication, tickets are issued by the TCC to the users. Tickets carry authorization information needed for the requested services. The billing information is stored in the TCC for the users to view and access after service provision.

The mechanism consists single signature and multi-signature schemes. A single signature can be used in tickets with only one bound entity (users or service providers). There are four roles in the single signature scheme, signer, verifier, credential_role and trusted_role. The credential_role in the TCC will issue tickets as well as provides information for the verifier to check the signature. Whether the signature is valid or not depends on the information. The trusted_role is a judge to solve the conflict between users and service providers.

The outline of the scheme is shown in Figure 2.2. In the system initialization, with SOAP methods, the trusted_role sends the private messages (r, S) to the signer when the signer I is set up, where r, S are computed by the trusted_role, will be used in the first verification by the credential_role and will be used as the first signature key by the signer. In the second step, the credential_role verifies if the data (I, r, D) sent by the signer are valid or not, where D is used in the ticket verification. The data

(I, D) will be put on a public directory in the TCC if the data are valid. At this time, the signer can complete a message signing job.

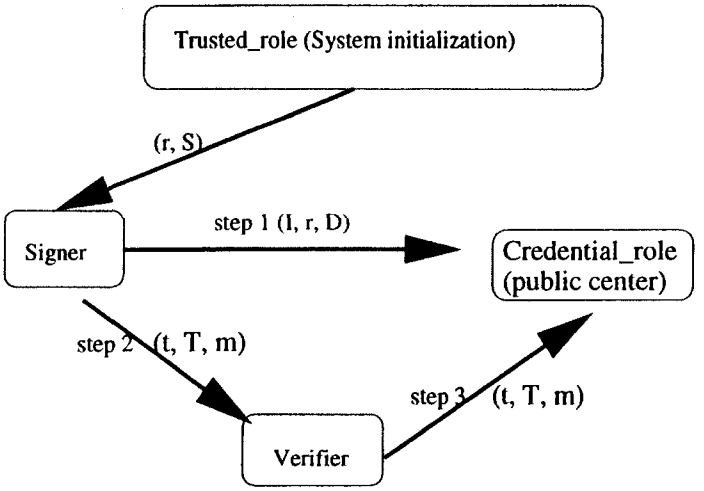

FIGURE 2.2 SINGLE SIGNATURE SCHEME

While the signer signs a message m, the signer will send the signed message (t, T, m) as a ticket to the verifier, and the latter checks if it is true or not, where t and T are computed by the signer and may include some service information and conditions, etc. The verifier cannot verify the message when the data (I, D) in the TCC are not correct. Then the credential_role can control the usage of the ticket, and even find who the signer is if it contacts the trusted_role. In the final step, the verifier sends a message which includes the ticket to the TCC while the ticket is true. The latter will update the data (I, D) that is used to issue a charging bill. The data (I, D) is changed while the ticket is used and the ticket is invalid if the verifier cannot get the correct data (I, D). Thus, the ticket cannot be used twice and the user can see a clear statement.

During the system initialization phase, the trusted_role computes a public composite modulus $n = pq$ where factors are big primes. The Trusted_role chooses also prime exponents and such that $e * d = 1(\mod (p-1)(q-1))$. The pair n, e made public, and d is kept secret by the TARC as the system key. The trusted_role computes

when the signer with identity I signs up: $r = k^e mod n, S = k * I(\mod n)$. Then, $D = S^e = r * I^e(\mod n)$. The trusted_role secretly sends (r, S) to the signer whose public identity is I. S will be used as the first signature key to issue a ticket. The signer with the public key I sends (I, r, D) to the credential_role, and the latter verifies the following equation: $D = r * I^e(\mod n)$. The credential_role publishes in a public directory the pair (I, D) for the signer with the public key I. The whole initialization process is outlined in Figure 2.3.

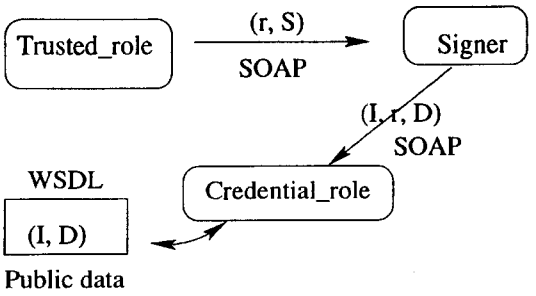

FIGURE 2.3 INITIALIZATION PHASE OF SINGLE SIGNATURE SCHEME

The verifier can access the public values n, e and the public pair (I, D) registered in the TCC. The data D in the TCC must be correct; otherwise the signed message (the ticket) cannot be verified by the verifier.

This paper also discussed the multi-signature scheme where the number of signers is not limited to two. The scheme is much more complicated than the single signature scheme. For both schemes, RSA public key cryptography is used to generate keys, signatures and verify signatures. Due to the focus of this thesis, we do not introduce details here.

In [7], an accountable anonymous access to services in mobile communication systems is proposed. The protocol demonstrates one-time use, post-paid tickets for one type of ticket. It consists a ticket acquisition protocol and ticket usage protocol. It is also based on Diffie-Hellman for agreement key exchange and public/private key generation.

In ticket acquisition protocol, the user U requests a ticket from the agency A:

$$U \rightarrow A: \quad (Uid, r_0)$$

where Uid is the identity of the user and r_0 is a freshly generated random number. The agency A constructs the ticket:

$$T = (sn, c_n, dh_u^+, K_u^+, Aid, Sig_{K_a^-}(sn, c_n, dh_u^+, K_u^+, Aid))$$

where sn is the serial number, dh_u^+ is the freshly generated public Diffie-Hellman key exchange parameter, K_u^+ is the freshly generated public signature verification key K_u^+, and c_n is the initial parameter for the "tick" protocol ($c_n = g^n(c_0)$, g is a one-way hash function). Aid is the agency's identity. The agency sends the ticket to the user together with the private parameters $c_0, dh_u^-, and K_u^-$:

$$A \rightarrow U: \quad \{T, c_0, dh_u^-, K_u^-, r_0\}_{K_{ua}}$$

The message is encrypted with the key K_{ua} using a symmetric encryption algorithm. K_{ua} is shared by the user U and the agency A.

In the ticket usage phase, the use computes a session key k using his private Diffie-Hellman key exchange parameter dh_u^- and the service provider's public parameter dh_s^+: $k = f_u(dh_u^-, dh_s^+)$. This key is used to authenticate the service provider and to encrypt and decrypt messages to and from the service provider.

The user sends a service request to the service provider:

$$U \rightarrow S: \quad (T, r_1)$$

where T is the ticket and r_1 is the freshly generated random number r_1.

The service provider computes the session key k, using its own private Diffie-Hellman key exchange parameter dh_s^- and the public parameter dh_u^+ of the user: $k = f_s(dh_s^-, dh_u^+)$. k is the same key as the one computed by the user, thus, it can be used to encrypt messages to the user.

The service provider authenticate itself and communicate the tariff t for the service by sending the following message to the user:

$$S \rightarrow U: \quad (t, h(k, t, r_1), r_2)$$

where r_2 is a freshly generated random number. The user can check the correctness of the hash value. Then, he confirms the service request by signing the serial number sn of the ticket T together with the tariff t and the random number r_2.

$$U \rightarrow S: \quad Sig_{K_u^-}(sn, t, r_2)$$

Then, the service provider verifies the signature and keeps it for later usage. The charging and payment is also based on the scheme in [3]. We assume that the last "tick" sent by the user is c_l. the last "tick" is always stored by the service provider.

1. The service provider sends a request for d ticks.
2. The user computes $c_{l-d} = g^{l-d}(c_0)$, and sends it to the service provider.
3. The service provider checks the "tick" by computing $g^d(c_{l-d})$. If the result is c_l, then it provides the next part of the service according to the tariff.

In short, the ticket based authentication mechanism proposed in [4–7] needs to run ticket acquisition protocol to get tickets. However, the ticket and the running of protocol require Diffie-Hellamn or RSA public key cryptography for key agreement

and digital signature. The computation complexity is very high. It is not practical for lower power computational mobile devices. In addition, in these works, a ticket cannot be reused. This results in the need for a quantity of tickets which is equal to the number of service requests.

2.3 UMTS Access Security

The access security features in UMTS are a superset of those provided in GSM. UMTS provides mutual authentication between the UMTS subscriber, represented by a smart card application known as the USIM, and the network.

UMTS specification has the following user identity confidentiality security features:

- *User identity confidentiality*: the property that the permanent user identity (IMSI) of a user to whom a services is delivered cannot be eavesdropped on the radio access link;
- *User location confidentiality*: the property that the presence or the arrival of a user in a certain area cannot be determined by eavesdropping on the radio access link;
- *User untraceability*: the property that an intruder cannot deduce whether different services are delivered to the same user by eavesdropping on the radio access link.

The design of the authentication and key agreement (AKA) protocol for UMTS reflects the results of an analysis of the threats and risks in GSM [8]. It was guided by the principle that the compatibility with GSM should be maximized. In particular, the changes to the GSM core network should be minimized.

The AKA procedure is the essence of authenticating a user to the network and vice versa. This is possible due to the pre-shared secret key K stored in the Authentication Center (AuC) and in the UMTS Subscriber Identity Module (USIM). The other parameters are derived from this key.

During an AKA procedure, messages with parameters to be confirmed by the User Equipment (UE), are delivered from AuC. Such parameters are joined together in an Authentication Vector (AV). The AV is delivered to the Core Network, which distributes parts of this AV through the access network to the UE. The UE must then perform some calculations to match this challenge. The result of the UE is sent back and checked against the AV where it originated. If the result matches, then the authentication is successful. If the result fails some other procedures are activated to correct the problem.

UMTS AKA achieves mutual authentication by the user and the network showing knowledge of a secret key K which is shared between and available only to the USIM and the AuC in the user's HE. In addition the USIM and the HE keep track of counters SQNMS and SQNHE respectively to support network authentication. The sequence number SQNHE is an individual counter for each user and the sequence number SQNMS denotes the highest sequence number the USIM has accepted.

2.3.1 Review of UMTS AKA

The method was chosen in such a way as to achieve maximum compatibility with the current GSM security architecture and facilitate migration from GSM to UMTS. The method is composed of a challenge/response protocol identical to the GSM subscriber authentication and key establishment protocol combined with a sequence number-based one-pass protocol for network authentication (see Figure 2.4).

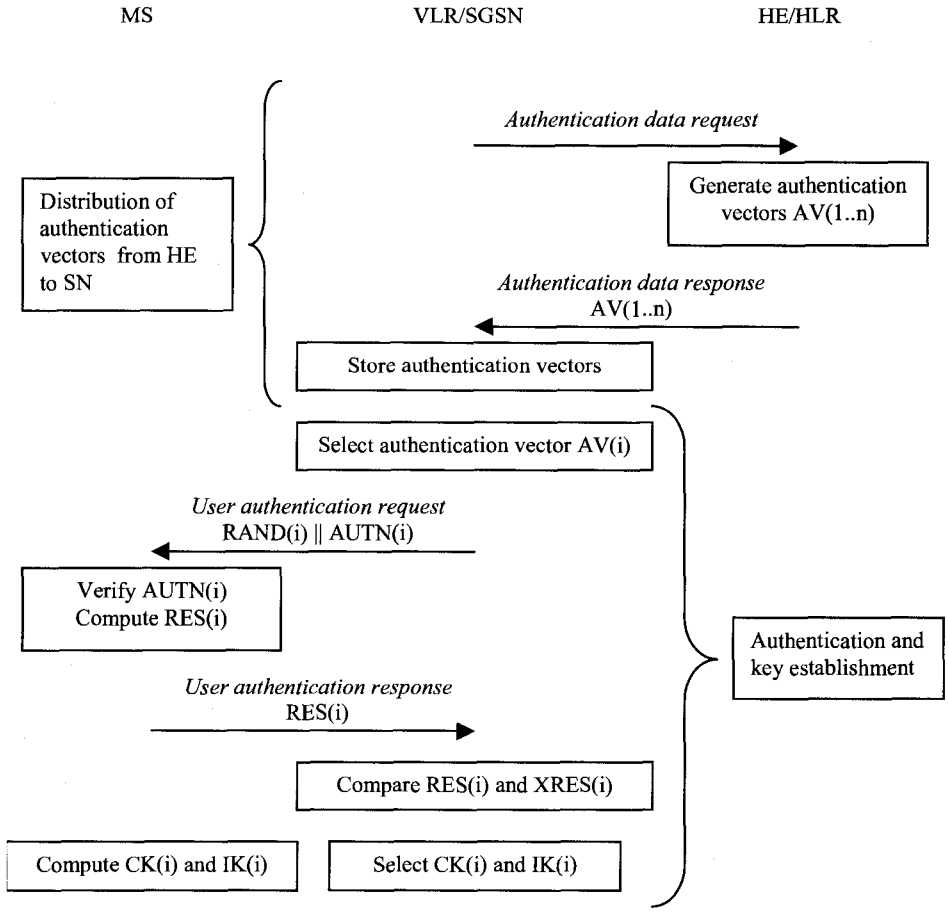

FIGURE 2.4 UMTS AUTHENTICATION AND KEY AGREEMENT

Authentication and key agreement consists of two procedures. First, the home environment (HE) distributes authentication information to the serving network (SN). Second, an authentication exchange is run between the user and the SN. The authentication information consists of the parameters necessary to carry out the authentication exchange and provide the agreed keys.

Upon receipt of a request from the VLR/SGSN, the HE/AuC sends an ordered array of n authentication vectors to the VLR/SGSN. The authentication vectors are ordered based on sequence number. Each vector consists of five components: a random number $RAND$; an expected response $XRES$; a cipher key CK; an integrity key IK; and an authentication token $AUTH$. Each vector is generated according to the

following steps.

1. HN generates a sequence number SQN from the counter SQN_{HN} and also generates an unpredictable random number $RAND$.

2. HN computes the following: $XRES = f2_K(RAND)$, $CK = f3_K(RAND)$, $IK = f4_K(RAND)$, $AK = f5_K(RAND)$,$MAC = f1_K(SQN\|RAND\|AMF)$.

3. HN assembles the authentication token $AUTH = SQN \oplus AK\|AMF\|MAC$ and $RAND, XRES, CK, IK, AUTH$.

4. HN increases SQN_{HN} by 1.

In each authentication vector, an authentication and key management field AMF is included, which serves to define operator-specific options in the authentication process.

The serving network SN invokes the second procedure by selecting the next unused authentication vector from the ordered array of authentication vectors. Each authentication vector is good for one authentication and key agreement between the mobile station and the serving network. The serving network SN sends to MS the random number $RAND$ and the authentication token $AUTH$ from the selected authentication vector.

Upon receipt of $RAND$ and $AUTN$, MS computes the anonymity key $AK = f5_K(RAND)$ and retrieves the sequence number $SQN = (SQN \oplus AK) \oplus AK$. Then MS computes $f1_K(SQN\|RAND\|AMF)$ and compares this with the MAC included in $AUTH$. If they are different, MS sends a user authentication reject message back to SN with an indication of the cause and abandons the procedure. Otherwise, MS verifies if the received sequence number SQN is in the correct range, i.e., $SQN > SQN_{MS}$. If MS considers the sequence number to be not in the correct

range, it sends a synchronization failure message back to SN. In this case, HN may need to resynchronize the counter SQN_{HN} maintained for the mobile station.

If the sequence number SQN is considered to be in the correct range, the authentication of the network is successful. In this case, MS computes $RES = f2_K(RAND)$ and sends it back to SN. Next, MS sets SQN_{MS} equal to SQN if $SQN_{MS} < SQN$. Lastly, MS computes the cipher key $CK = f3_K(RAND)$ and the integrity key $IK = f4_K(RAND)$.

Upon receipt of the user authentication response, SN compares RES with the expected response $XRES$ from the selected authentication vector. If RES is equal to $XRES$, the authentication of the user is successful and SN selects the cipher key CK and the integrity key IK from the selected authentication vector. If RES and $XRES$ are different, SN sends an authentication failure report to the HN and abandons the procedure.

If a resynchronization is required, a synchronization failure message sent by MS includes a resynchronization token $AUTS$ in the form $AUTS = Conc(SQN_{MS}) \| S_{MAC}$, $Conc(SQN_{MS}) = SQN_{MS} \oplus f5^*_K(RAND)$, $S_{MAC} = f1^*_K(SQN_{MS} \| RAND \| AMF)$. Upon receipt of the synchronization failure message, SN sends $RAND$ and $AUTS$ to the home network with an indication of the synchronization failure.

After receiving the synchronization failure indication, HN retrieves SQN_{MS} and verifies whether the value of SQN_{MS} mandates that SQN_{HN} needs to be changed, i.e.,$SQN_{HN} < SQN_{MS}$. If necessary, HN also verifies the correctness of the S_{MAC} included in the resynchronization token $AUTS$ and sets SQN_{HN} equal to SQN_{MS}. Subsequently, HN sends a new batch of authentication vectors to SN. When SN receives a new batch of authentication vectors, it deletes the old authentication vectors stored for the user.

2.3.1.1 Weaknesses and Attacks

The weaknesses and corresponding solutions of UMTS AKA are widely discussed in literature [9], [10], [11]. One of the big problems is the employment of sequence number introduced in the AKA. The generation, allocation, verification, and management of the sequence number is a complicated matter [10]. It is really a tough task to maintain the synchronization. Several scenarios which could result in synchronization failures are discussed in [9],[10]. Once a synchronization fails, the re-synchronization procedure involves quite a few principals and network domains [8]. Moreover, a possible crush in the database which storing the sequence number will cost a lot to re-establish the synchronization [9].

In short, the sequence number adopted in UMTS AKA brings extra tremendous work over normal operations and potential attacks. To prevent synchronization and re-synchronization of the sequence number, one good solution is to abandon completely the sequence number as shown in [9], [10], [11].

UMTS AKA was well designed by by the European Telecommunications Standards Institute (ETSI). As other security protocols, UMTS AKA provides confidentiality, integrity, mutual authentication (CIA). It also defends against man-in-the-middle and replay attacks. Nevertheless, two attacks are shown in [9]: *redirection attack*, *active attack in corrupted networks*. The redirection attack occurs when an adversary entices a legitimate mobile station to camp on the radio channels of the false base station. Since the authentication vector can be used for any serving network, an adversary can intercept the vector and impersonate both the mobile user and the serving network. As a result, the adversary can redirect user traffic to an unintended network.

The active attack occurs while a network is corrupted and an adversary could forge an authentication data request from the corrupted network to obtain authentication vectors. In addition, the adversary could force the sequence number to be set in a

high value by flooding the authentication data to the home network. As a result, the adversary can start an active attack against legitimate users.

2.3.2 Related Works

In [9], an adaptive AKA (AP-AKA) is proposed (see Figure 2.5). As the synchronization and re-synchronization of sequence numbers are really complicated, the sequence number is totally discarded in the protocol. Instead, the MS keeps a list of unused vectors' indexes to verify that an authentication vector is really sent from the serving network and was not used before. This helps to prevent replay attack.

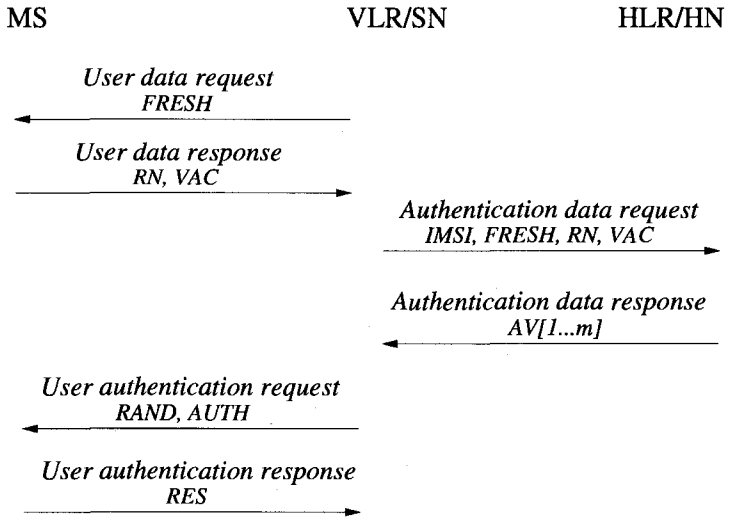

FIGURE 2.5 AP-AKA

In AP-AKA, each mobile station and its home network share an authentication key K and three cryptographic algorithms F, G, and H, where F and H are message authentication codes and G is a key generation function. In practice, the authentication key is usually generated by the home network and programmed into the mobile station during service provisioning. Unlike in 3GPP AKA, however, the home network in AP-AKA does not maintain a dynamic state, e.g., the counter, for each individual

subscriber.

The mobile station can verify whether an authentication vector was indeed requested by a serving network and was not used before by the serving network. The protocol AP-AKA specifies a sequence of six flows. Each flow defines a message type and format sent or received by an entity. Depending on the execution environment, entities have the flexibility of adaptively selecting flows for execution.

The AP-AKA differentiate two ways of execution depending on whether SN has unused authentication vectors. If SN does not have unused authentication vectors for the user, all the six flows will be carried out. SN starts the protocol by sending a user data request to the user. A random number, denoted by $FRESH$, is included in the request. After receiving the request, the MS generates a random number RN and computes a message authentication code, denoted by $VAC = F_K(FRESH \| RN \| ID_{SN})$, where ID_{SN} denotes the identity of SN. The mobile station then sends a user data response back to including RN and VAC. Subsequently, SN sends an authentication data request to the home network, including $IMSI, FRESH, RN$, and VAC. Upon receipt of the authentication data request from SN, HN retrieves the secret key K of the user and verifies the correctness of the received VAC. If the verification succeeds, HN generates a batch m of authentication vectors and sends them back to SN. Each authentication vector consists of four components $(RAND, XRES, SK, AUTH)$ and is indexed by an integer $idx, 1 \leq idx \leq m$. The index number describes the order of the authentication vector in the batch. To generate an authentication vector, HN proceeds as follows:

1. HN allocates an index number idx for authentication vector and generates a random number $RAND$;

2. HN computes the following: the expected response $XRES = F_K(RAND)$; a session key $SK = G_K(RAND)$; and two message authentication codes $RN_{idx} =$

$H_K(idx\|RN)$, and $MAC = F_K(RAND\|idx\|RN_{idx})$;

3. HN assembles the authentication token $AUTH = idx\|RN_{idx}\|MAC$ and the authentication vector $(RAND, XRES, SK, AUTH)$.

After receiving the authentication vectors from HN, SN takes out one of them from the batch and stores the rest in its database. Then it sends a user authentication request to the mobile station, including $RAND$ and $AUTH$ from the selected authentication vector. After receiving the user authentication response from the mobile station, SN compares if $RES = XRES$. If not, the authentication of the user fails and SN abandons the connection. Otherwise, the user authentication is successful and the agreed session key is the SK from the selected authentication vector, where SK may be a concatenation of a cipher key and an integrity key.

If SN has unused authentication vectors for the user, it selects an unused authentication vector from its database and starts the protocol by executing the fifth flow, i.e., sending a user authentication request to the mobile station, including $RAND$ and $AUTH$ from the selected authentication vector. After receiving the user authentication response, SN compares if $RES = XRES$ and proceeds as described earlier.

The main contribution of AP-AKA is that it discards the sequence numbers and replace with a predetermined indexes shared between the mobile station and the home network. If ever the mobile station loses the indexes, it needs only to ask the SN to request another array of vectors and no resynchronization procedure to be executed between the MS and the HN.

The protocol also differentiates the scenarios that the MS stays in his home network or roams to a serving network. In each scenario, the execution of protocol is different. One disadvantage of this approach is the bandwidth consumption and storage space

of authentication vectors [11].

In [10], the authors employ two techniques: a one-time password/hash chaining technique [12] and keyed-Hash Message Authentication Code (HMAC) [13] instead of the sequence number to establish mutual authentication (see Figure 2.6).

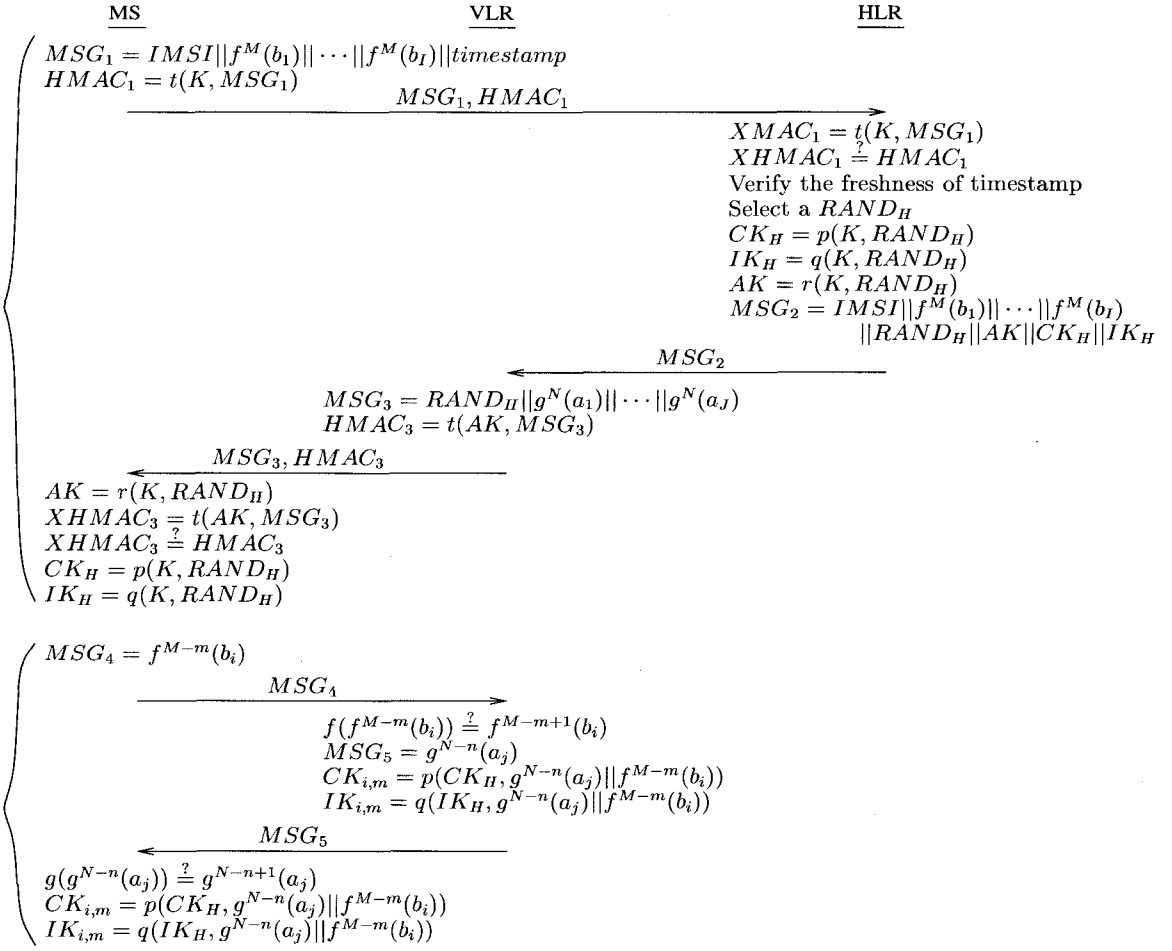

FIGURE 2.6 ENHANCED REGISTRATION AND AKA

When a MS roams to a VLR, he generates a hash chain and sends $MSG_1, HMAC_1$ to his HE/HLR. After the HLR verifies the authenticity of MSG_1, it then prepares MSG_2 to send to the VLR. In order for the MS to verify the authenticity of the VLR later on in the AKA phase, MSG_3 and $HMAC_3$ is prepared and sent by the VLR.

Thus, each MS and VLR pair has two distinct sets of hash chains for mutual authentication for future. Since each authentication uses one chain position, the MS can prove its identity to the VLR at most $I \times M$ times, whereas the VLR to the MS $J \times N$ times. Within each set of hash chains, it can be agreed that the chain with lower id is used.

One of the big advantages of the hash chain is that there is no synchronization of sequence numbers needed between entities. The two hash chains can be used only and always for one MS and VLR pair even the MS roams out and roams back. This means the MS has to keep all the pairs of hash chains for every visited VLR. This would impose extra storage space for the mobile station.

Another minor disadvantage is the employment of timestamps for message freshness while the MS sends his hash chain to the HE/HLR. However, to guarantee freshness by timestamps, both MS and HE/HLR must keep local clocks periodically synchronized by a reliable source of time in a secure manner. Between synchronization with the reliable time source, local clocks may drift [14]. Both entities must allow a time window for timestamps to compensate for local clock drift and the fact that messages take time to cross a network. Moreover, the mobile user might have his own "personalized clock", for example, 3 minutes in advance. Thus, it is not reasonable to urge the mobile user to keep an exact clock. To avoid the use of timestamps, the MS can run extra protocol steps, for example, by exchanging random numbers, before sending his hash chain.

The timestamps problem exists also in X-AKA proposed by [11]. In this extended AKA, the mobile user sends his message authentication code based on the timestamp, and the temporary key generated is also a function of the timestamp. Nevertheless, one advantage of this approach is the introducing of the temporary key between the MS and the VLR. Through this key, both entities can authenticate each other subsequently without the involving of sequence numbers.

2.4 DoS Attacks in Mobile Networks

Denial-of-service (DoS) is becoming a growing concern in communication networks. The function of a DoS attack is fundamentally to flood its target servers with numerous connect requests to prevent them from being accessible to any other legitimate users' requests or providing services. Consequently, the target server becomes too overburdened to respond to requests from its legitimate users and attackers. As a result, it has insufficient system resources to respond to legitimate requests on the network. Usually, the DoS attack packets are indistinguishable from the legitimate packets, which makes detection and traffic filtering difficult.

Broadly speaking the attacks can be of three forms.

- Attacks exploiting some vulnerability or implementation bug in the software implementation of a service to bring that down.
- Attacks that use up all the available resources at the target machine.
- Attacks that consume all the bandwidth available to the victim machine.

DoS attacks may be either destructive or degradative. A destructive DoS attack prevents the availability of a service completely. In a degradative (non-destructive) DoS attack the performance of a service is reduced, such as in a flooding attack overloading a network link or a host CPU. This will typically cause only temporary problems, and a system will recover automatically as soon as an attack terminates.

In modern mobile communication networks the danger of DoS attacks is threatening different targets: specific services, the communication infrastructure and the mobile devices. The current UMTS system architecture and protocols are designed to accommodate Internet-like services, while it also exposes the UMTS network to various Internet attacks. Communication interruption caused by Internet DoS attacks is one

of the big concern. For example, the UMTS public land mobile network (PLMN) can be easily congested and therefore paralyzed by bogus traffic from the Internet. Additionally, the increasing complexity of mobile devices can coordinate to launch typical DoS attacks, taking advantage of limited resources in the network, such as the radio spectrum, etc.

2.4.1 Client Puzzle Technique

One mechanism to prevent DoS is client puzzles. Client puzzles [15–20] are a promising technique that aims to provide service guarantees to legitimate clients. A client puzzle is a small cryptographic problem created by the server in response to a client request. The client should first commit its resources to solving the puzzle before completing the remaining part of the communication protocol. An example of client puzzle is to solve the square root of $2 = \sqrt{2} \times \sqrt{2}$. The client has to find the answer in a brute force manner, while validating the answer is much easier to the server.

Clients get access to a service only after they prove their legitimacy. For each service request, the client is forced to solve a cryptographic "puzzle" before the server commits its resources. This imposes a large computational task on adversaries generating traffic in large quantities. The main idea behind client puzzles is that any client requesting service must allocate some of its own resources (processing time or memory) before the server commits its resources for the connection. This defends against attacks staged by a large number of zombie computers using authentic IP addresses.

The client puzzle does not make resource consumption attacks impossible, but it makes them more expensive for the attacker in the sense that successful attacks require considerably more resources from the attacker. Moreover, by varying the complexity of the puzzle, the cost of an attack can be varied too; this allows one

to adjust the system according to the assumptions he has about the strength of the attacker.

A good puzzle should have the following properties [15]:

1. Creating a puzzle and verifying the solution is inexpensive for the server.
2. The cost of solving the puzzle is easy to adjust from zero to impossible.
3. The puzzle can be solved on most types of client hardware (although it may take longer with slow hardware).
4. It is not possible to precompute solutions to the puzzles.
5. While the client is solving the puzzle, the server does not need to store the solution or other client-specific data.
6. The same puzzle may be given to several clients. Knowing the solution of one or more clients does not help a new client in solving the puzzle.
7. A client can reuse a puzzle by creating several instances of it.

There are several variants of puzzle-based anti-DoS protocols. The puzzle auctions protocol of Wang and Reiter [18] requires the client to solve a puzzle of a certain difficulty level before it can initiate a session with the server. The request message must be accompanied by the solution to the puzzle. In the approach proposed by Feng *et al.* [16], instead clients increasing the puzzle difficulty level for each server rejection, the server is required to respond with a puzzle of the current highest difficulty level. The server allocates resources only if it receives the correct solution from the client. By adapting the puzzle difficulty level in proportion to the current load, the server can force clients to solve puzzles of varying difficulty.

2.4.2 Related Works

Time-lock puzzles are based on the notion that a client must spend a particular amount of computation time performing repeated squaring [19]. Another puzzle mechanism is to force clients to reverse one-way hashes calculated at the server given the original random input with n bits erased [17]. In order to vary the difficulty level, n is either increased or decreased. The client performs a brute-force search on the erased bits by hashing each pattern in the space until it finds the answer. A different manner based on hash function is to make use of partial collision of two hash values to form puzzles instead [15]. The client is forced to solve the k-bit partial collision. The hashed value of the solved puzzle must be k-bit (the most significant bits) partial collided with all 0's string. The difficulty level is determined by the number of collision bits. A partial collision is that two hashed values have partially identical strings. In this puzzle protocol (Figure 2.7), the server generates nonce N_S and the difficulty level k to the client.

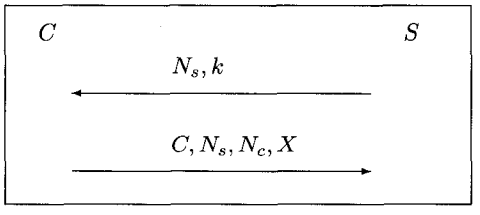

FIGURE 2.7 PUZZLES BASED ON PARTIAL COLLISIONS

The client generates his own nonce N_C. The necessity of N_c is to prevent an attacker to solve the solution based on N_s and send to the server before the client does. The client spends its time to find a required partial collision and from the following equation and sends the solution (C, N_s, N_c, X) to the server.

$$h(C, N_s, N_c, X) = \overbrace{000\cdots00}^{k \text{ bits } 0\text{'s}} \underbrace{x_1 x_2 \cdots}_{\text{other bits}}$$

h	=	a hash function (MD5 or SHA1)
C	=	the client identity
N_s	=	the server's nonce
N_c	=	the client's nonce
X	=	the solution of the puzzle
k	=	the difficulty level of the puzzle
$000\cdots00$	=	the k first bits, must all be 0
$x_1x_2\cdots$	=	other bits, garbage data

Another approach, called Completely Automated Public Turing test to tell Computers and Humans Apart, or CAPTCHA, is a program that can distinguish between humans and computers. First widely used by Alta Vista to prevent automated search submissions, CAPTCHAs are particularly effective in stopping any kind of automated abuse, including DoS attacks. They work by presenting some test that is easy for humans to pass but difficult for computers to pass; therefore, they can conclude with some certainty whether there is a human on the other end. One example of distorted words is shown in Figure 2.8.

FIGURE 2.8 PUZZLES BASED ON PARTIAL COLLISIONS

2.5 Discussion

We have given a review of current literature related to our proposed research objectives. All of the contributions are very interesting and we would like to consider the improvements based on them. As our main research objective is to design a set of

lightweight security mechanisms in order to meet with the constraints of mobile networks, we would like to design our own mechanisms which use lightweight techniques.

There are two major common disadvantages for the current contributions for mobile service access: expensive cost due to public key cryptography and non-reusable tickets. In order to reach our goal, we need to replace the public key cryptography techniques with lightweight techniques. Without a doubt, to ensure the security, the whole mechanism should be carefully redesigned. Considering the unbalanced characteristic in mobile network, we may forward heavy calculation to the server side. We may also consider the possibility to make a ticket reusable. A hash-chain might help.

UMTS AKA has been widely studied and the current contributions eliminate the synchronization of sequence numbers between the mobile station and the home network. Thus, the security has been enhanced as well as the overall performance. In all these mechanisms, however, the authentication vector can be used once and only once. And it is always the home network to generate the authentication vectors for all its subscribers. We might consider to improve the performance further through a way to more efficiently use the vectors. Of course, one of the serious attack, replay attack, must be avoided.

As defending against DoS attacks, the current client puzzle reviewed in this chapter uses a hash function problem which must be resolved by the client. An obvious issue is that there is a difficulty level gap between two neighbor levels. If we could find a way to decrease the gap, the performance would be much better.

In the following chapters of this thesis, we are going to present our designed mechanisms related to these areas. Our goal is to eliminate the concerns that we just mentioned above.

CHAPTER 3

A LIGHTWEIGHT MOBILE SERVICE ACCESS BASED ON REUSABLE TICKETS

3.1 Introduction

The rapidly developing mobile technologies widely enhance the abilities of mobile networks to offer new services. Third-generation (3G) mobile systems such as Universal Mobile Telecommunications Service (UMTS) bring mobile users a broad range of new value-added services (VASs). To access a service, users must authenticate themselves with the value-added service provider (VASP), while the latter must also be authenticated to ensure it is not malicious. For mobile access across multiple service domains, the traditional access mechanisms [8–11, 21–23] require the exchange of authentication information between the home domain and the foreign domain using roaming agreements. This may involve complicated and expensive activities over large scale networks and limit future mobile applications.

Given the low-power computing capability of mobile devices, designing of secure service access mechanisms which are suited for wireless mobile networks is a nontrivial challenge. Due to the complexity of the underlying problems, public key cryptography (PKC) or asymmetric cryptography usually involves operations such as modular multiplication and exponentiation, which are much more computationally expensive than operations in symmetric cryptography. Hence, it is best to move demanding computational requirements from the mobile device to a powerful server in protocol design.

In mobile communications, a ticket is a piece of data which is analogous to tickets

we use in various social events. The ticket proves that the user has been authorized to access to VASs like tourism guide. Thus, there is no need to run long distance protocols with an on-line third credential party of the accessing user in order to validate the user's certificate. Ticket-based service access is an appealing approach in mobile communication systems where users roam and contact VASPs in foreign domains [2, 4–7, 24].

Users do not have to disclose personal information, when ticket authentication is required. In addition, tickets can be used in different VASPs, which is of great importance in heterogeneous mobile networks. For example, a ticket-based mechanism [2, 4–7, 24], together with a trusted agent which issues tickets and handles payments on users' behalf, can protect the user privacy.

Due to its flexibility, the ticket idea is also widely used for access control in dynamic and distributed networks. In [25], a ticket-based access control in mobile ad-hoc networks is introduced to meet the dynamics of network topology and node membership. Mobile nodes without a valid ticket are classified as misbehaving and denied from any network access. Here, a ticket acts as a passport for a networking node. It provides a simple yet effective mechanism for controlling the access of mobile nodes.

In [26], a ticket-based secure delegation service is proposed to meet the requirements of distributed computing. It supports multiple domain models by extending the Kerberos [21] framework. By using tickets based on PKC, this mechanism realizes flexible key management and reliable session key generation between the client and the provider.

Other ticket-based applications can also be found in literature. A ticket-based address resolution protocol (TARP) [27] implements security by distributing centrally generated MAC/IP address mapping attestations, called tickets, to clients as they join the network. In [28], the authors proposed a secure wireless LAN access based

on an authentication ticket which preserves clients' privacy.

This chapter proposes a novel ticket-based service access mechanism which incorporates two major contributions compared to various works. First, the proposal is lightweight on the mobile user's side. High computational complexity is transfered from the mobile user side to the ticket server and the VAS provider (VASP). Second, the ticket is reusable. Two hash chains are introduced: an authentication chain and a payment chain. The authentication chain allows a ticket to be reusable as well as achieving mutual authentication and non-repudiation. The payment chain is a staircase incremental value hash chain which is lightweight, practical and may avoid chain reinitialization.

This chapter is organized as follows. Section II covers the background work and concepts. The proposed service access model and protocols are shown in Section 3.2 where ticket acquisition, service provision, payment and billing, ticket revocation and reuse issues are considered. Section 3.4 illustrates BAN logic [29] proofs and security analysis. In Section 3.5, the proposed mechanism is compared with related work including classical mechanism Kerberos and existed mobile service mechanisms. Finally, conclusions are given in Section 3.6.

3.2 Background Work and Motivation

3.2.1 Ticket Types vs. Service Types

We encounter different types of tickets in our social life. In [5, 6], the authors list four types of tickets (see Table 3.1). Some tickets bind a given user to a specific service provider. Certain tickets, however, do not restrict users and service providers. Certain tickets can only be used once (e.g., fight ticket), while other tickets can be reused many times.

TABLE 3.1 TICKET TYPES

Ticket Types	t_0	t_1	t_2	t_3
User	-	-	+	+
VASP	-	+	-	+

As for their methods of payment, tickets can be categorized as prepaid and postpaid. The t_0 type ticket is the simplest: it looks like cash. Since it does not bind to any principal, it must be prepaid. A t_1 type of ticket means that the same ticket can be used by different users that it can be lent to someone else. If not free, most t_1 tickets are prepaid like prepaid cellphone cards or concert tickets issued by a VASP, and are accepted by the same provider. Otherwise, the VASP could not find the right user for refunds. The ticket usage is quite simple: the user buys a ticket from a kiosk, scratches it and inputs the PIN number to the VASP. Then, the VASP grants the service access to the user as long as there is money left in the ticket.

Generally speaking, using t_2 and t_3 tickets, is not as simple. The VASP must be paid from the legitimate ticket holder. Hence, a postpaid ticket is usually found to be a t_2, t_3 ticket although the latter can also be prepaid. A postpaid ticket works much the same as a credit card granting the service access right and requiring payment after the service has been delivered. Only when a mobile user starts his service, a potential payment occurs and both the VASP and the user sign the payment. An aggregated invoice is generated and sent to the user much like the balance statement from the credit card companies. The ticket validation requires authentication from both the ticket holder and the VASP. If it is prepaid, the payment is made immediately from the money left in the ticket.

Traditionally, most mobile systems entirely rely on an implicit trust relationship between a user and a VASP. In the future, mobile systems will become larger and

there may co-exist many VASPs. Such a trust model is not suitable anymore because a VASP could not be completely trusted. It could even be malicious to mobile users in order to increase its profit. Hence, in order to access a service, mutual authentication is required and the potential payment after the service provision should be non-repudiable. Non-repudiation means that the user cannot deny the service usage; the VASP cannot forge a payment. Thus, ticket binding to the user is required. In this situation, t_2, t_3 type tickets are of greater interest. The remainder of this paper will only consider t_2, t_3 type tickets.

While service access is controlled by a ticket, billing and charging is determined by the service characteristics. There are different types of services. Considering the variety of mobile VASs, we can classify them into two categories: entertainment and information. The entertainment includes mobile movies, mobile games, TV-related services, etc. The information category includes other services such as location-based services (LBS), mobile marketing, mobile email and weather forecast. Service fees can be charged according to usage time, traffic or the number of access times. Some services (such as personal mobile banking) can be accessed free for those with the appropriate access right (such as the owner of an account). We list these types of services in Table 3.2.

TABLE 3.2 SERVICE TYPES

Service Types	**Charge Metrics**
s_0	free, access right is required
s_1	the number of access times
s_2	service provision time
s_3	quantity of traffic

Knowing the type of service is helpful to design the charging and billing scheme during the service provision. Charging is very simple for s_0 and s_1 service. For s_2 and

s_3 services, the scheme should provide continuous charging instead of pre-charging or post-charging. Pre-charging is not suitable as it is difficult to predict the usage time or quantity of traffic. Post-charging is risky as users can terminate the service deliberately just before the charging occurs.

3.2.2 Ticket Usage

The main advantages of ticket-based access are the following [7]:

- **Flexibility**. A mobile user can choose personalized services by acquiring corresponding tickets and does not require long term contract with VASPs;
- **Scalability**. A ticket can contain all of the necessary information for service provision. Long distance protocol running between the VASP and the user's home domain is not necessary;
- **Privacy**. A ticket does not contain the user's personal information. It only proves the user as a legitimate ticket holder.

Although these advantages make ticket-based access an appealing approach for service provision, several issues regarding the security of ticket design and usage must be kept in mind. One problem shown in [24] is duplication. There are two cases: a legitimate user deliberately makes copies of an acquired ticket; an eavesdropper steals a ticket. For both cases, a secure authentication associating with the ticket is required in order to prevent illegal use of a ticket.

Another issue is forgery [24]. Only the ticket server can construct a valid ticket. Forgery can be prevented through encryption and digital signature from the ticket server.

In this paper, we will also address the ticket reuse issue. The current ticket-based service access mechanisms do not allow ticket reuse. This implies that a mobile user must acquire as many tickets as the number of service requests. Thus, the gain of no long distance cross domain authentication is not well worth only one service request, although ticket have privacy and flexibility advantages. If a ticket is reusable, it will be much more interesting.

3.2.3 Lightweight Non-Repudiation Techniques

We have claimed that non-repudiation is required for the service access. Because of the limited computation capability of mobile devices, the non-repudiation technique should be lightweight at the mobile user side. Digital signature is widely used for non-repudiation. However, it is usually computationally expensive. A one-way hash function chain was first proposed by [30]. Hash functions are generally faster than digital signature algorithms. Hash chains, a useful cryptographic technique, are used to provide services like authentication [30, 31], micropayments [3, 32–36], signatures [37–40], multicasting [41] and certificate revocation [42]. Let $f(x)$ be a one-way hash function and a hash chain of a length of N is constructed by applying N times of hashing:

$$f^{(N)}(x) = f(f(\cdots f(x)\cdots))$$

Hash values are called *nodes* and $f^{(N)}(x)$ is called the tip. We define $f^{(0)}(x) = x$. With knowledge of $f^{(N)}(x)$, $f^{(N-1)}(x)$ cannot be generated without the knowledge of x. However, given $f^{(N-1)}(x)$, its correctness can easily be verified through $f^{(N)}(x)$, simply checking if it satisfies the equation: $f^{(N)}(x) = f(f^{(N-1)}(x))$. This property comes directly from the property of one-way hash functions. Once the tip is used for verification, the hash chain is reduced and the new tip becomes $f^{(N-1)}(x)$.

A cryptographic message authentication code (MAC) is a short piece of information

used to authenticate a message. The input of a MAC function consists of a secret key and a message to be authenticated:

$$MAC = h(Key, Message)$$

where h is a one-way hash function. The MAC technique is widely used in UMTS authentication and key agreement (AKA) [8]. The MAC value protects both a message's integrity as well as its authenticity, by allowing verifiers (who also possess the secret key) to detect any changes to the message content.

The combination of MAC and hash chain can provide the non-repudiation purpose. A claimant randomly selects a nonce, generates a hash chain and its MAC, and sends both to the verifier. The hash chain can be used to prove the claimant for N times. At the first authentication, the claimant sends $f^{(N-1)}(x)$ to the verifier. The verifier checks if it satisfies the equation $f^{(N)}(x) = f(f^{(N-1)}(x))$. If it does, the verifier updates the stored value to $f^{(N-1)}(x)$ for the following authentication. The property of the hash chain prevents non-legitimate users from impersonating the legitimate user. Obviously, it also guarantees that the legitimate sender of a message cannot subsequently deny having sent the message.

3.3 Service Access through Tickets

3.3.1 Service Access Model

Cases where a ticket is issued and validated by the same service provider will not be covered as they do not belong to the typical models in cross domain authentication. We define the user's home domain as the ticket issuer (ticket server) and his foreign domain as the VASP. The proposed model contains four principals: *a certification authority (CA)*, *a ticket server (S)*, *a mobile user (U)* and *a VASP (P)* as illustrated

in Figure 3.1.

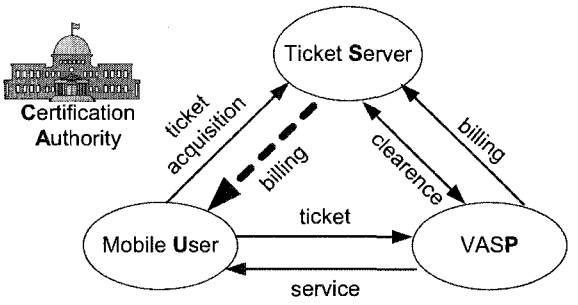

FIGURE 3.1 SERVICE ACCESS MODEL

The last three principals all subscribe to the CA which is responsible for issuing certificates to mobile users and generating key pairs (a public key and a secret key) to the ticket server and the VASP. The mobile user obtains a ticket through presenting his certificate to the ticket server. This is realized by a ticket acquisition protocol. After successfully validating the certificate, the ticket server issues a ticket to the mobile user. Here, the mobile user obtains the right to access a service and does not need to pay the ticket. Once the mobile user gets the ticket, he can use it to request the service from the VASP anytime, anywhere.

The VASP needs to access the ticket server occasionally, for example, for billing. In fact, in traditional cross domain authentication in GSM and UMTS [8, 22], the home domain is always accessible for the foreign domain. For every time of service provision, the VASP generates a billing record and sends it to the ticket server. The ticket server collects all the billing records and generates an aggregated bill to the user periodically. The user pays the bill to the ticket server. Finally, the VASP gets paid through the financial clearance with the ticket server.

3.3.2 Ticket Acquisition

As we analyzed previously, in mobile networks, the users usually have limited power mobile devices to communicate with more powerful servers like VASPs. Although mobile devices become increasingly powerful today, public key cryptographic operations (encryption/decryption) remain still two or three orders of magnitude slower than symmetric key cryptography operations. It is preferable to move high computational complexity from mobile devices side to the ticket server side as much as possible.

We denote U the mobile user, S the ticket server and P the VASP. We abuse the uppercase letters for principals and their identities. Before running the protocol, U first obtains a certificate $Cert_U^S$ and his long term symmetric key K_U from the CA. The certificate is given as:

$$Cert_U^S = \langle E_S(U, K_U, expire, Udata, Sig_{CA}(U, K_U, expire, Udata)) \rangle$$

where K_U is U's long term symmetric secret key. This key, however, has a special property compared with ordinary secret keys. Instead of being known only by U, the key can also be known by the ticket server during the ticket acquisition phase. Aside from issuing certificates, the CA is responsible for generating symmetric keys for mobile users. In this certificate, Sig_{CA} is the secret signing algorithm of the CA. The result after running the signing algorithm is the CA's digital signature. The digital signature of a message is a string which depends on this message and on the CA's secret key, in such a way that anyone can check the validity of the signature through the CA's signature verification algorithm. The verification algorithm takes a message and its corresponding public key to compute the signature.

Inside the certificate, U is the mobile user's identity and the field $Udata$ includes all other relevant information: version number, serial number, user ID, issuer ID

and issuer name, etc. All the three fields with the signature are encrypted with the ticket server's public key ($E_S(X)$ denotes the public key encryption under S's public key). This means that U possesses a certificate specific to the ticket server S before communicating with S. A similar server specific certificate can also be found in [43].

Once the mobile user has his certificate in hand, he can start his authentication to the ticket server and obtain a ticket. The protocol is illustrated as follows:

$$\begin{aligned} U \rightarrow S: &\quad Cert_U^S, \{R_1, \cdots, R_T\}_{K_U} \\ S \rightarrow U: &\quad \{R_1 \oplus \cdots \oplus R_T, id, ticket, C_1, \cdots, C_T, K_{UP}\}_{K_U} \end{aligned}$$

The protocol starts when U sends a set of random nonces $R_1, \cdots, R_T$ encrypted by his long term symmetric key K_U ($\{X\}_K$ denotes symmetric encryption of X under key K) and the certificate of U under S. These nonces are used to generate a set of hash chains. The length of the hash chain N is predetermined in advance for both principals.

The ticket server decrypts the certificate with its private key, then verifies the signature using the CA's public signature verification algorithm. When S checks successfully the certificate from U, it discovers the secret key K_U and stores its relevant information such as the expire date. It obtains the nonces and computes the corresponding hash chains.

Depending on the different types of service, if it can be known in advance, the ticket server generates payment chains of different lengths. Otherwise, it generates a payment chain based on experiences. It chooses a set of random nonces $C_1, \cdots, C_T$ to generate payment hash chains $g^{(M)}(C_t)$ (g is a one-way hash function, $1 \leq t \leq T$) with length of M. Then, it generates an authentication key K_{UP} for future authentication between U and P, together with the *ticket* and its unique *id*. The ticket server encrypts all of these fields with K_U and sends them back to U. The ticket is defined

as:

$$\begin{aligned} ticket = \; & \langle E_P(P, K_{UP}, id, f^{(N)}(R_1), \cdots, f^{(N)}(R_T), g^{(M)}(C_1), \cdots, g^{(M)}(C_T), data, \\ & Sig_S(P, K_{UP}, id, f^{(N)}(R_1), \cdots, f^{(N)}(R_T), g^{(M)}(C_1), \cdots, g^{(M)}(C_T), data))\rangle \end{aligned}$$

The ticket is specific to the VASP P. It is encrypted with P's public key, which means that only P can decrypt it and the ticket belongs to a t_3 type ticket. This requires that the ticket server knows the VASP's public key through the CA. The signature of S in the ticket is used to prevent a fake ticket. The *id* is the unique identity of the ticket and the *data* field includes all detailed ticket information: issuer ID, issuer name, issued time, service type, service name, times of access, etc.

When the ticket is sent, the ticket server stores the information pertaining to the ticket for future billing verifications. Here is an example:

id	P	K_U	$expire$	$R_1, \cdots, R_T$	$C_1, \cdots, C_T$	$valid$

where *expire* is the K_U expire date. The nonces are reserved for future authentications and payment chain validation. The field *valid* is reserved by the ticket server. When a ticket needs to be revoked, this field is set to FALSE. Otherwise, it is set to TRUE. When U receives the message, he checks the correctness of $R_1 \oplus \cdots \oplus R_T$. If it is correct, he keeps the *ticket* for future service request. Otherwise, he discards the *ticket*.

3.3.3 Service Provision

Once U obtains a valid ticket, he can provide it to P for service provision. The protocol is illustrated as follows:

$$\begin{aligned} U \rightarrow P: &\quad ticket, \{f^{(N-i)}(R_t), P, N_U\}_{K_{UP}} \\ P \rightarrow U: &\quad \{f^{(N-i+1)}(R_t), N_U + 1, N_P, T_P\}_{K_{UP}} \\ U \rightarrow P: &\quad \{N_P \oplus T_P\}_K \end{aligned}$$

The protocol starts when U wants to validate his ticket to get access to the service from P. He sends the *ticket*, together with the hash chain (the (i,t)-th time, $1 \leq i \leq N$, $1 \leq t \leq T$), the identity of P and a nonce N_U encrypted with the authentication key K_{UP} previously generated by the ticket server S during the ticket acquisition phase. Here, the inclusion of P is to tell P that U knows P as his peer communication principal.

When P receives the service request, it decrypts the *ticket* with its private key and checks its validity with S's signature verification algorithm. Then, it compares this signature to the one received. If they are identical, the ticket is considered as valid. Then, it gets the *id* of the ticket and searches its database to see if the ticket is used for the first time. If so, it stores the hash chains and the tip $f^{(N)}(R_1)$; otherwise, it ignores the chains. It gets the key K_{UP} and decrypts the second part of the message. It checks if the hash chain $f^{(N-i)}(R_t)$ satisfies the equation:

$$f(f^{(N-i)}(R_t)) = f^{(N-i+1)}(R_t)$$

If so, it updates the hash chain by removing the current tip $f^{(N-i+1)}(R_t)$. The hash chain becomes shorter and the new tip is $f^{(N-i)}(R_t)$. Note that the mobile user uses the hash nodes between $f^{(N-1)}(R_t)$ and $f^{(0)}(R_t)$ (inclusive) for authentication

while the VASP uses the hash nodes between $f^{(N)}(R_t)$ and $f^{(1)}(R_t)$ (inclusive) for verification. After that, the t-th chain is abandoned and the $(t+1)$-th chain is used until all the hash chains run out.

Then, it chooses a nonce N_P and generates a fresh session key $K = f3_{K_{UP}}(N_U \oplus N_P)$, where $f3$ can be defined as in UMTS authentication and key agreement [8]. It sends back $f^{(N-i+1)}(R_t)$ as confirmation, together with $(N_U+1), N_P$ and the timestamp T_P encrypted by the shared key K_{UP}.

When U receives the message, he decrypts the message and checks the correctness of $f^{(N-i+1)}(R_t), (N_U+1)$. If they are correct, he knows that mutual authentication has been realized. He computes the fresh session key $K = f3_{K_{UP}}(N_U \oplus N_P)$ and sends back $\{N_P \oplus T_P\}_K$ for agreement of T_P as the service start time.

3.3.4 Ticket Acquisition via VASP

In the proposed model, a mobile user acquires a ticket before a service request. It implies that the mobile user knows the VASP which he will request service from, before roaming into the service domain. This means, the ticket is type t_3. However, it is not always the case. Sometimes, the mobile user cannot predict or simply has no knowledge of the VASP, or he might need only a t_2 ticket. In this situation, the user can request a ticket from the VASP ignoring the VASP's identity. The VASP acts as a proxy to pass the request to the ticket server. This action is also helpful when the ticket server is not accessible directly by a mobile user even the user wants a t_3 ticket. The following protocol describes this action. In this case, it becomes the traditional cross domain authentication as in GSM and UMTS [8, 22].

$$
\begin{aligned}
U \rightarrow P: \quad & Cert_U^S, \{R_1, \cdots, R_T\}_{K_U} \\
P \rightarrow S: \quad & E_S(Cert_U^S, \{R_1, \cdots, R_T\}_{K_U}, N_P, \\
& Sig_P(Cert_U^S, \{R_1, \cdots, R_T\}_{K_U}, N_P)) \\
S \rightarrow P: \quad & E_P(N_P + 1, ticket, \{R_1 \oplus \cdots \oplus R_T, id, ticket, C_1, \cdots, C_T, K_{UP}\}_{K_U}, \\
& Sig_s(N_P + 1, ticket, \{R_1 \oplus \cdots \oplus R_T, id, ticket, C_1, \cdots, C_T, K_{UP}\}_{K_U})) \\
P \rightarrow U: \quad & \{R_1 \oplus \cdots, id, ticket, C_1, \cdots, C_T, K_{UP}\}_{K_U}, \{\{R_1, \cdots, \}_{K_U}, N_P, T_P\}_{K_{UP}} \\
U \rightarrow P: \quad & \{N_P \oplus T_P\}_K
\end{aligned}
$$

The mobile user sends the certificate and a set of nonces encrypted with the secret key K_U to the VASP P. The latter appends a nonce N_P, signs the whole message and sends it to the ticket server S. When S successfully checks the validity of the signature and the certificate, it generates the hash chains based on the nonces received and billing chains $g^{(M)}(C_1), \cdots, g^{(M)}(C_T)$. Then, it creates a *ticket* and computes the confirmation $N_P + 1$ and $\{R_1 \oplus \cdots \oplus R_T\}_{K_U}$. It signs the whole message and sends it to P.

When P receives the message, it validates the signature. If it is valid, it keeps the *ticket* and gets the shared key K_{UP}. It forwards to U the corresponding part. It encrypts the $\{R_1, \cdots, R_T\}_{K_U}$ with the shared key K_{UP} for confirmation, together with a nonce N_P and the timestamp T_P. The nonce and the timestamp have the same meaning as in the service provision protocol. They are used to generate fresh session key and service start time agreement.

3.3.5 Payment and Billing

The payment chain $g^{(M)}(C_1), \cdots, g^{(M)}(C_T)$ included in the ticket are used by the VASP to charge the service. However, the usage of the payment chain is different

from the usage of the authentication chain. For each time service provision, the user takes off a new node from the authentication chain while restarting the same (only one) payment chain from the beginning. This means the payment is only related to the current service provision and independent to the previous service provision.

During the service provision, payments happen several times depending on different types of service. The idea of "tick" payments based on hash chains was first introduced by Pedersen [3] and has been widely adopted in [7,32–35]. The payment in this paper is also based on ticks. The charging request is repeated several times according to the service provision requirements. The following steps are based on those in [7]. We assume that the last "tick" sent by the user is c_j.

1. The VASP sends a request for d ticks according to the expense;
2. The user sends back the response: the tick $c_{j-d} = g^{(j-d)}(C_t)$ with the computed message authentication codes $MAC'_{j-d} = h(K_{UP}, (P, T_j, c_{j-d}, f^{(N-i)}(R_t)))$ and $MAC_{j-d} = h(K_U, (P, T_j, c_{j-d}, f^{(N-i)}(R_t)))$, where T_j is the timestamp;
3. The VASP checks if $c_j = g^{(d)}(c_{j-d})$. If so, it continues the service provision; otherwise, it stops.

Even if the service is free (s_0 type), the VASP still asks the user to sign the payment in order to prevent the user denying the service. In this case, the value of first tick can be defined as zero.

Although the user can refuse to release a tick after a period of service, he cannot gain too much (only the value after the last tick c_j). When the service terminates, the VASP generates a billing record and subsequently sends the proof of the service

provision to the ticket server.

$$P \rightarrow S: \quad E_S(id, P, T_P, T_j, c_j, f^{(N-i)}(R_t), MAC'_j, MAC_j, \\ Sig_P(id, P, T_P, T_j, c_j, f^{(N-i)}(R_t), MAC'_j, MAC_j))$$

This billing message contains the ticket id, the identity of the VASP, the service start time and end time, the last tick from the user c_j, the current authentication chain and MAC'_j, MAC_j. The message is authenticated by the user and signed by the VASP. Both of them cannot deny the service provision. The ticket service checks the validity of the payment chain and MAC'_j, MAC_j. If there is already a billing record corresponding to the current chain of this ticket, it is rejected by the ticket server it might be a replayed billing by the VASP. Otherwise, a billing record is stored and an aggregated bill will be sent to the user.

A problem concerning the length of the payment chain, however, should be considered. Due to its finite length, the system has to restart when it runs out the hash chain. Simply generating a longer hash chain will not help out fundamentally because the length of the hash chain cannot be set too large in practice. A simple solution is to run again the service provision protocol to use the next node of the authentication chain. Another way is to re-initialize the old hash chain. In [44], the authors proposed a method which applies public key cryptography recursively to generate infinite length hash chains. In [45, 46], the authors introduced a notion of "tying" multiple finite length hash chains through one time signature. All of these methods increase resource consumption.

In this paper, we introduce a new payment method which may avoid hash chain reinitialization and keep the hash chain in a reasonable length. The idea is illustrated as follows. We do not try to prolong the hash chain while trying to use every tick efficiently. Traditionally, every tick has the same value. We define a *staircase incremental value function* $value(j)$ associating with each tick according to its position in

the hash chain (the position of the tick $g^{(j)}(C_t)$ is j).

$$value(j) = \begin{cases} a & \text{if } (n-j) \leq l_0, \\ a \cdot p_1 & \text{if } l_0 < (n-j) \leq l_1, \\ a \cdot p_2 & \text{if } l_1 < (n-j) \leq l_2, \\ \cdots & \cdots \\ a \cdot p_k & \text{if } l_{k-1} < (n-j) \leq n. \end{cases}$$

where $a \geq 1$ and $1 < p_1 < p_2 < \cdots < p_k$. The hash chain is cut into different segments and the ticks in different segments are associated with different values. Thus, the hash chain becomes a staircase. The ticks in the first stair ($(n-j) \leq l_0$) is the *cheapest*, while those in the last stair ($l_{k-1} < (n-j) \leq n$) are the most expensive. For example, the value of one single tick in the second stair can be equivalent to the value of two ticks of the first stair. For example, $p_1, p_2, p_3, \cdots$ can be $2, 3, 4, \cdots$ or $2^1, 2^2, 2^3, \cdots$. Intuitively, the ticks in the back are less frequently used than those in the front during the charging procedure.

A more detailed example is as follows. Suppose a user needs to spend 10 dollar for his service. Each tick represents 10 cent in a regular hash chain. Suppose that a regular hash chain is only 50 in length, thus the total value of the hash chain is 5 dollar. Obviously, the hash chain needs to be reinitialized. In our scheme, we may define the first 10 ticks in the hash chain as the same value as in regular hash chain. However, we define the other ticks in more expensive values. For example, each tick between 11st and 20th represents 20 cent; each tick between 21st and 30th represents 30 cent; each tick between 30st and 40th represents 40 cent; each tick between 41st and 45th represents 60 cent; each tick between 46th and 50th represents 80 cent. Thus, the total value will become 17 dollar and hash chain does not need to be reinitialized. Moreover, as ticks become more and more expensive, they will be used less and less.

As a result, the VASP requests less and less the ticks from the user. Thus, our scheme improves the system performance.

The value function is agreed among the principals in advance. If it is possible to know the type of service in the ticket acquisition phase, the payment chain can be tailored to the service. For s_0 and s_1 service, one tick is sufficient. For s_2 and s_3 service, the length of the chain and the value function needs to be carefully designed. Through this way, we may avoid reinitializing the payment hash chain. This approach is not perfect for payment, but practical and lightweight.

The mico-payment chain was proposed in [3], and the security has been proved. Our proposed staircase incremental value function is essentially the same scheme from the security aspect. We change only the value function assigned to the payment chain. Actually, we can say that the payment chain in [3] is a special case where the staircase incremental value function defines the same cost for all its ticks.

3.3.6 Ticket Revocation and Reuse

The ticket revocation issue is simple. If a ticket is considered to be revoked (e.g., the mobile user's secret key K_U expires or the mobile user notices the ticket server to revoke a ticket), the ticket server sends $REVOKE$ message to the corresponding VASP:

$$S \rightarrow P: \quad E_P(id, REVOKE, Sig_S(id, REVOKE))$$

The VASP checks if the ticket is already stored (used by the ticket holder before). If not, it means that the ticket has been issued and has not been used yet; the VASP stores the revoked ticket information. Otherwise, it updates the status of the ticket REVOKED.

When a user requests a service with a ticket, the VASP checks if the ticket has been revoked. If so, the service request is rejected.

In the solution described in this paper, a ticket can be reused, which is different from other ticket-based mechanisms. This is realized and secured through the hash chains $f^{(N)}(R_1),\ f^{(N)}(R_2), \cdots, f^{(N)}(R_T)$ shared between the mobile user and the VASP. This hash chain can be used $N \times T$ times. After that, this ticket becomes invalid and should be abandoned. Thus, we can say the ticket is reusable within a certain limit.

3.4 System Security Analysis

3.4.1 Logical Proof by BAN Logic

The goal of a secure access mechanism is to ensure that authentication is realized for all principals involved in a communication. The authentication protocol relies on network relationships. Any failed relationship will allow an adversary to masquerade as legitimate principals in runs of the protocol. Belief logics can be used to examine trust relationships in security protocols. We use BAN logic [29] to analyze the protocols in this thesis.

This chapter only presents proofs from the ticket acquisition protocol and the service provision protocol. The ticket acquisition via VASP is similar and we do not show its proof. For convenience, we put two protocols together. The idealized protocol is as follows:

M1. $U \rightarrow S: \quad E_S(\{\overset{K_U}{\longmapsto} U\}), \{R\}_{K_U}$

M2. $S \rightarrow U: \quad \{R, E_P(\{U \overset{K_{UP}}{\longleftrightarrow} P\}), U \overset{K_{UP}}{\longleftrightarrow} P\}_{K_U}$

M3. $U \rightarrow P: \quad E_P(\{U \overset{K_{UP}}{\longleftrightarrow} P\}), \{R, N_U\}_{K_{UP}}$

M4. $P \rightarrow U: \quad \{R, N_U, N_P, T_P\}_{K_{UP}}$

M5. $U \rightarrow P: \quad \{N_P \oplus T_P\}_K$

The **M1.**, **M2.**,...,**M5.** represent the message rounds. In order to make it easy to be understood, we list the basic notations and inference rules used in this thesis below. For further information, please consult the original paper [29].

- $\overset{K_U}{\longmapsto} U$: U has K_U as a public key.
- $E_S(X)$: X is encrypted by S's private key.
- $U \overset{K_{UP}}{\longleftrightarrow} P$: U and P may use the shared secret key K_{UP} to communicate. The key K_{UP} is good, in that it will never be discovered by any principal except U and P, or a principal trusted by either U or P.
- $P \overset{X}{\rightleftharpoons} Q$: The formula X is a secret known only to P and Q, and possibly to principals trusted by them. Only P and Q may use X to prove their identities to one another. An example of a secret is a password.
- P **believes** X: P believes X, or P would be entitled to believe X. In particular, the principal P may act as though X is true. This construct is central to the logic.
- **fresh**(X): The formula X is fresh; that is, X has not been sent in a message at any time before the current run of the protocol. This is usually true for nonces, that is, expressions invented for the purpose of being fresh.
- P **controls** X: P has jurisdiction over X. The principal P is an authority on X and should be trusted on this matter. For example, a server is often trusted to generate encryption keys properly. This may be expressed by the assumption that the principals believe that the server has jurisdiction over statements about the quality of keys.
- P **sees** X: P sees X. Someone has sent a message containing X to P, who can read and repeat X (possibly after doing some decryption).

- P **said** X: P once said X. The principal P at some time sent a message including the statement X. It is not known whether the message was sent long ago or during the current run of the protocol, but it is known that P believed X then.

- The *message-meaning* rule concerns the interpretation of messages. That is, if P believes that the key K is shared with Q and sees X encrypted under K, then P believes that Q once said X.

- The *nonce-verification* rule expresses the check that a message is recent and, hence, that the sender still believes in it. That is, if P believes that X could have been uttered only recently (in the present) and that Q once said X (either in the past or in the present), then P believes that Q believes X.

For sake of simplicity, we replace $Cert_U^S$ and the *ticket* by $\{\overset{K_U}{\longmapsto} U\}_{K_S}$ and $\{U \overset{K_{UP}}{\longleftrightarrow} P\}_{K_P}$ respectively and ignore signature verification fields inside. The hash chain and the i-th hash node are also simplified as R. The verification of hash chain is treated independently outside of the protocol, and the returned R implies the correctness of the verification.

To analyze this protocol, we first state the assumed initial beliefs of the principals:

U	**believes**	$\mathbf{fresh}(R)$	S	**believes**	$\mathbf{fresh}(R)$
U	**believes**	$\overset{K_U}{\longmapsto} U$	S	**believes**	$\overset{K_S}{\longmapsto} S$
S	**believes**	$\mathbf{fresh}(K_{UP})$	S	**believes**	$U \overset{K_{UP}}{\longleftrightarrow} P$
U	**believes**	$\mathbf{fresh}(N_U)$	P	**believes**	$\mathbf{fresh}(N_U)$
S	**believes**	$\mathbf{fresh}(N_U)$	P	**believes**	$\overset{K_P}{\longmapsto} P$
U	**believes**	$(S$ **controls** $U \overset{K_{UP}}{\longleftrightarrow} P)$	P	**believes**	$(S$ **controls** $U \overset{K_{UP}}{\longleftrightarrow} P)$
P	**believes**	$\mathbf{fresh}(N_P)$	U	**believes**	$\mathbf{fresh}(N_P)$
P	**believes**	$\mathbf{fresh}(T_P)$	U	**believes**	$\mathbf{fresh}(T_P)$

Both U and P trust the shared key K_{UP} generated by S and believe its freshness. All principals believe the freshness of R, nonce N_U and timestamp T_P. Each principal trusts its own secret key.

As receiving **M1**, the principal S has S **sees** $E_S(\{\overset{K_U}{\longmapsto} U\}), \{R\}_{K_U}$. In other words, the principal S can decrypt the message and read the secret key K_U and the random number R. With the hypothesis S **believes** $\overset{K_S}{\longmapsto} S$, the message-meaning rule yields S **believes** U **said** $(\overset{K_U}{\longmapsto} U)$. In other words, S believes that it is U, not anyone else, who sent the message. With the belief U **believes** $\overset{K_U}{\longmapsto} U$, we can deduce S **believes** U **said** (R) . This means that S believes that it is U, not anyone else, who sent the random number R. The initial belief S **believes** **fresh**(R) and the nonce-verification rule yield S **believes** U **believes** **fresh**(R) . In other words, both S and U believe that the random number R is fresh.

Similarly, as receiving **M2**, we have U **believes** $U \overset{R}{\rightleftharpoons} S$, U **believes** $U \overset{K_U}{\rightleftharpoons} S$. In other words, both U and S know the secrets R and K_U. Since we have hypothesis S **believes** $U \overset{K_{UP}}{\longleftrightarrow} P$, we obtain U **believes** $U \overset{K_{UP}}{\longleftrightarrow} P$. This means that U believes that K_{UP} is the shared secret key between U and P.

The deduction of **M3** is similar to **M1**. We do not show the details here. The final result from **M3** is P **believes** U **believes** **fresh**(N_U) . This means that N_U is fresh and P believes this.

While U receives the **M4**, with the result U **believes** $U \overset{K_{UP}}{\longleftrightarrow} P$, the message-meaning rule for shared keys applies. As a result, we can have the following result: U **believes** P **said** (R, N_U, N_P, T_P) . Moreover, there are the following hypotheses: U **believes** **fresh**(N_P) , U **believes** **fresh**(T_P) . The nonce-verification rule applies and yields U **believes** P **believes** (R, N_U, N_P, T_P) . In other words, both P and U believe the freshness of R, N_U, N_P, T_P.

At this point, U and P are mutually authenticated. The last message, **M5**, is specifically used for both the timestamp T_P and the new session key agreement. It does not influence the beliefs just established above.

We summarize the above formal analysis in natural language as follows. As the certificate is encrypted by the ticket server S and signed by CA, only S can verify it through its private key and CA's signature verification algorithm. Then, S can read U's secret key and further decrypt the random numbers. When U receives the ticket from S, he can know that it is sent from S because the K_U is only known by U and S.

As the ticket is encrypted by service provider's public key, only P can decrypt it. When P receives this ticket, it can get the shared secret key K_{UP} stored in the ticket and thus decrypt the hash node and the random number N_U in the message. At this point, both U and P believe that K_{UP} is a shared secret key between them. They trust each other and use this key for secret communication. The next step is to verify the hash chain stored in the ticket. Of course, its security is guaranteed by K_{UP}. Finally, both U and P contribute a random number and generate alone the session key K.

3.4.2 Security Analysis

It has been suggested that the BAN logic is unable to deal with security flaws resulting from interleaving of protocol steps [47]. We analyze the security aspects of the proposed protocols by considering possible attacks.

3.4.2.1 Forge Tickets

Outside attackers cannot fabricate or forge tickets, because they cannot generate a valid signature in the name of the CA. In the ticket acquisition phase, since the K_U is encrypted in the certificate, only S can verify U's certificate $Cert_U^S$ and thus know U's long term secret symmetric key K_U and retrieves the nonces. Moreover, the *ticket* should contain a signature from S. Only S can generate the signature. The signature is also generated based on the nonces from U.

3.4.2.2 Mutual Authentication

The freshness of the ticket implies the freshness of the authentication chain and the payment chain inside. During the service provision phase, U sends a two-part message: *ticket*, the encrypted current hash node and a nonce N_U. The *ticket* is encrypted by P's public key and contain P's identity indicating that P has the right to validate the *ticket*. The *ticket* is signed by the ticket server and P can check if the ticket is valid through the signature contained in the *ticket*.

When P receives the *ticket*, it cannot justify if the *ticket* owner is a legitimate user. It decrypts the hash chain and the nonce with the K_{UP} retrieved from the *ticket*. Then, it sends back the updated hash chain, the nonce and its challenge (a nonce N_P) encrypted by K_{UP}. When U receives this message and validates it successfully, U knows that P is the right service provider. In other words, P is authenticated by U. The message that U sends back to P contains the response. When P validates successfully the response, P knows that U is the right *ticket* owner. In other words, U is authenticated by P. Thus, mutual authentication is reached.

3.4.2.3 Replay and Man-in-the-middle Attacks

Although an attacker can eavesdrop a valid *ticket*, he cannot use it later, because he cannot get the K_{UP} and the hash chain encrypted in the *ticket*. Only P can decrypt the *ticket* and get the hash chain then. An attacker cannot modify the *ticket* and the encrypted hash chain. Even if the attacker replays the encrypted hash chain, he cannot be authenticated by P as P will fail to validate the hash chain. Thus, replay and man-in-the-middle (MITM) attacks are prevented.

3.4.2.4 Key Freshness

The authentication key K_{UP} is used only during the authentication and known only by U and P. The new session key is generated for each time service provision and valid only at the current session. Both U and P contribute their nonces to establish this key. Thus, key freshness is guaranteed.

3.4.2.5 Malicious Service Provider

In mobile networks, a service provider could be malicious and deliberately leak a user's ticket to someone else. In the payment and billing protocol, U signs the ticks used with his private key K_U which can only be verified by the ticket server. This signature comprises the current tips of the authentication chain and the payment chain; nobody can tamper and generate it. Note that the payment is also agreed and signed by the VASP. Even if the VASP is malicious and helps an eavesdropper get a valid ticket, the eavesdropper cannot use it because he cannot generate a valid payment without the long term secret key K_U. Consequently, S will refuse the payment and P gets nothing. Note that the payment is also required for s_0 type service (free) although the value is zero. This is helpful for the ticket server to solve possible conflicts between

the user and the VASP.

3.4.3 Discussion

We designed a complementary protocol to help a mobile user acquire a ticket via a VASP. When U needs to acquire a ticket via a VASP, it is mandatory that the ticket server be accessible. This is similar to Kerberos [21] in which the server should always remain on-line. However, the situation of our proposal is less serious because this protocol would not happen frequently. It can be mitigated by using multiple ticket servers. It seems that the system executes cross domain authentication and loses the benefit of a ticket. However, since the ticket described in this thesis is reusable, one time cross domain authentication is worth future reused local domain authentication.

The authentication chain allows a ticket to be reused $N \times T$ times. Nevertheless, ticket reuse does not mean a duplicate ticket is acceptable because it would fail in authentication without a correct hash node. In order to use this chain correctly, both the user and the VASP have to synchronize its current position in the chain. If the synchronization fails (e.g., the user loses the position information), it is difficult for them to resynchronize. Consequently, the ticket should be abandoned. As for the payment chain, if it is not well designed, it can be run out before the service terminates. Experiments are helpful to determine the length of the chain and the value function.

3.5 Comparisons with Related Work

3.5.1 Comparison with Kerberos

Kerberos [21, 48], a computer network authentication protocol developed by MIT to protect network services provided by Project Athena, allows individuals communicating over an insecure network where individuals prove their identities to one another in a secure manner.

Kerberos (Version 5) protocol is based on the Needham-Schroeder protocol [49], but makes use of timestamps as nonces. It makes use of a trusted third party, called a Key Distribution Center (KDC), which consists of two logically separate parts: an Authentication Server (AS) and a Ticket Granting Server (TGS). Kerberos works on the basis of tickets which serve to prove the identity of users.

We can specify Kerberos protocol as follows, where Alice (A) authenticates herself to Bob (B) using a server (S).

$$
\begin{aligned}
A \rightarrow S: &\quad A, B \\
S \rightarrow A: &\quad \{T_S, L, K_{AB}, B, \{T_S, L, K_{AB}, A\}_{K_{BS}}\}_{K_{AS}} \\
A \rightarrow B: &\quad \{T_S, L, K_{AB}, A\}_{K_{BS}}, \{A, T_A\}_{K_{AB}} \\
B \rightarrow A: &\quad \{T_A + 1\}_{K_{AB}}
\end{aligned}
$$

Server S stands for both AS and TGS, and K_{AB} stands for the session key between A and B. In this protocol, $\{T_S, L, K_{AB}, A\}_{K_{BS}}$ acts as a ticket that the client authenticates himself to B. In order to confirm B's identity and its recognition of A, A sends $\{A, T_A\}_{K_{AB}}$ as an authenticator and B replies with $\{T_A + 1\}_{K_{AB}}$.

The protocol security relies heavily on timestamps T_S, T_A and lifespans L as reliable indicators of the freshness of a communication. Thus, one drawback is that all

principals must keep local clock synchronization. Between two synchronizations with the reliable time source, local clocks may drift [14]. Participants must allow a time window for timestamps to compensate for local clock drift and the fact that messages take time to cross a network. In Kerberos, all tickets are time-stamped and have limited lifetimes. Ideally, no authentication ticket remains valid longer than the time an expert hacker would need to crack the encryption. The value of L should be carefully set in order to prevent replay attack by an eavesdropper. Authentication tickets are session-specific to improve the security of the system by ensuring that no authentication ticket remains valid after a given session is complete. In other words, a ticket can be used only once. Whenever the user wants to access the service again, he should apply a new ticket. In other words, the ticket cannot be reused.

The security of the proposed proposal in this thesis, however, relies on a shared hash chain instead of timestamps. Thus, the validity of a ticket can last long time and can be reused. This will mitigate the burden of the ticket server to issue tickets.

While the original Kerberos does not provide clients anonymity, one of the basic features of tickets is user privacy during service provision. Recently, an Internet draft [50] proposed a mechanism to allow a client to request an anonymous ticket.

Moreover, Kerberos does not consider the payment and billing issue. If detailed billing records are needed by users, one solution is to add a record into B's database for each time of service by A. However, since both A and B only realize mutual authentication, A has no way to authorize B to generate a real record. The ticket here can be generated by B easily and B claims that A did access these services. It is also possible that B deliberately leaks the secret key and the ticket to help eavesdroppers to impersonate A. Similarly, A can also deny the service that he really did.

Payment and billing activities are carefully designed through an improved payment

hash chain described in this thesis. Only U can sign the payment and nobody can forge this signature. Replaying a ticket to steal money by a malicious VASP is defended; denying the service from both principals are also impossible.

As for scalability, Kerberos is limited to university-sized communities as the encryption specifications are designed primarily for conventional secret key cryptography. This results in a potential limitation in Kerberos scalability. Shared symmetric keys must be established and maintained between every user and the KDC, and between every application server and the KDC.

This issue is quite different in the mechanism of this thesis. Although there is a secret authentication key shared between the ticket server and the client, this key is actually not established directly between the client and the ticket server. Instead, it is generated by the CA and transferred to the ticket server, by the client on his certificate. The ticket server does not directly have to maintain the secret key as it can obtain this key from the client's certificate while the client presents his certificate for ticket request. In addition, the key is stored with each ticket only for payment verification. Similarly, the client and the VASP do not have to maintain the authentication key either since the authentication key can be easily obtained from the ticket for each time of service request.

3.5.2 Comparison with Extended Kerberos

As indicated by Neuman et al. [51], the traditional Kerberos presents an attractive security target in the form of the KDC which maintains a shared symmetric key with every principal. In the event of a KDC compromise, all the symmetric keys will be divulged to the attacker and will have to be revoked [52].

Various studies have been done to extend Kerberos with PKC [51–54]. These ex-

tensions integrate PKC into the initial authentication exchange and distribute most of the authentication workload away from the trusted intermediary to the communicating parties. Generally speaking, PKC can simplify the distribution of keys in the Kerberos environment because secret-key cryptology (SKC) requires a separate secret key to be shared between every pair of users. Thus, both the client and the KDC have a public-private key pair in order to prove their identities to each other over the open network.

The extended Kerberos improves the scalability and security. Nevertheless, as with the traditional Kerberos, the extended Kerberos still relies on timestamps for freshness indicators. Consequently, the timestamped ticket can only be valid in a single session. The introduction of PKC also increases the computation requirement for clients. The billing issue remains the same.

3.5.3 Comparisons with Mobile Service Mechanisms

Many ticket-based mobile service access mechanisms [2, 4–7, 24] have been proposed to address the VAS access issue in mobile networks. We compare them with the work in this thesis.

A secure billing for mobile information services conducted by Advanced Security for Personal Communications Technologies (ASPeCT) in UMTS is described in [2]. There are three principals in this pre-paid charging scheme: mobile user, service provider and UMTS service provider. This work relies on a third trusted party (TTP) to support mobile telecommunications security services which use PKC. When requiring services from the VASP, the user pays for the service by sending micropayment tokens ("ticks") as shown in [3]. The VASP is able to check the validity of these ticks with a user's credential certificate issued by the UMTS service provider, or an appropriate TTP acting on behalf of the UMTS service provider. The UMTS service provider

will collect the billing information from VASPs and in turn forward to the user. A two phase charging protocol is illustrated. In the first phase, the user and the VASP authenticate each other and agree on a Diffie-Hellman session key. The second phase is responsible for data transfer through user's ticks.

This work has several differences comparing to the work of this thesis. First, the protocol in [2] is based on PKC and the computational complexity is more expensive than symmetric cryptography operations. This requires powerful mobile devices. In this thesis, the mobile user needs only operations of hash and symmetric encryption/decryption. Second, the protocol focuses on authentication between the users and the VASP, and disregards different kinds of services in service provision. Third, the mobile user has to send his identity to the VASP. Thus, the VASP is constantly aware of the identity of users accessing services.

In [7], an accountable anonymous access to services in mobile communication systems is proposed. It is also based on Diffie-Hellman PKC and composed of three roles: user, service provider and customer care agency. The protocol demonstrates one-time use, post-paid tickets for one type of ticket. Its charging is also based on ticks [3]. There are two major differences between this mechanism and the solution proposed in this thesis. First, the user has to generate a session key using Diffie-Hellman method. It involves a high level computational complexity. Second, the ticket cannot be reused. Although the user can obtain a few tickets once for later use, it is possible that a user would run out of his tickets while roaming. In this case, the user could not access a service since the customer care agency could not be reached.

In [4], the authors present a ticket model which includes three principals: the mobile user, the service provider, and the ticket server. In this model, the mobile user is assumed to register with the ticket server. This service access mechanism can be split into two phases: ticket acquisition and ticket validation. The disadvantages of this work are similar to that in [2, 7]: a high-level computational complexity and

non-reusable tickets. The authors did not discuss the payment and billing issue.

Another similar service access mechanism [24] is based on prepaid tickets that can only be used with the service provider issued them. Payment and billing issue is not discussed. The service provider can replay a ticket and users has no way to prove they have not used it.

In [5, 6], the authors propose another ticket-based model which involves a Trusted Credential Center (TCC) and a Trusted Authentication and Registration Center (TARC). In both mechanisms, different ticket types are considered with a single or multiple signatures required for ticket validation and the prevention of ticket reuse. All tickets cannot be modified by the users. Ticket duplication, including reusing a ticket and transferring a ticket, are discussed. One of the disadvantages is the high level of computational complexity required for the signer and the verifier who could be a mobile user. The signature schemes are based on RSA which is based on large prime numbers factorization. In the single signature scheme, the signer picks a large random number and runs large exponent operations twice while the verifier has to run it three times. All of these operations require users have powerful devices. The situation is even more serious when multiple signatures are involved. As with other ticket-based mechanisms, tickets cannot be reused.

We simulated our mechanism and the mechanisms proposed in [4–7]. There are four ticket protocols in this simulation: 1 for our protocol, 2 for the protocol in [5,6], 3 for the protocol in [4] and 4 for the protocol in [7]. We chose 128 bits for symmetric key size and 1024 bits for public key size. The ticket and the certificate have the size of 64 bytes. We chose Blowfish algorithm for symmetric encryption/decryption. For all the same message fields, we used the same size for all protocols. For protocol 1, we set the hash chain size 12. The message size comparison is shown in the Figure 3.2.

In this figure, the acquisition message means all the messages involved during the

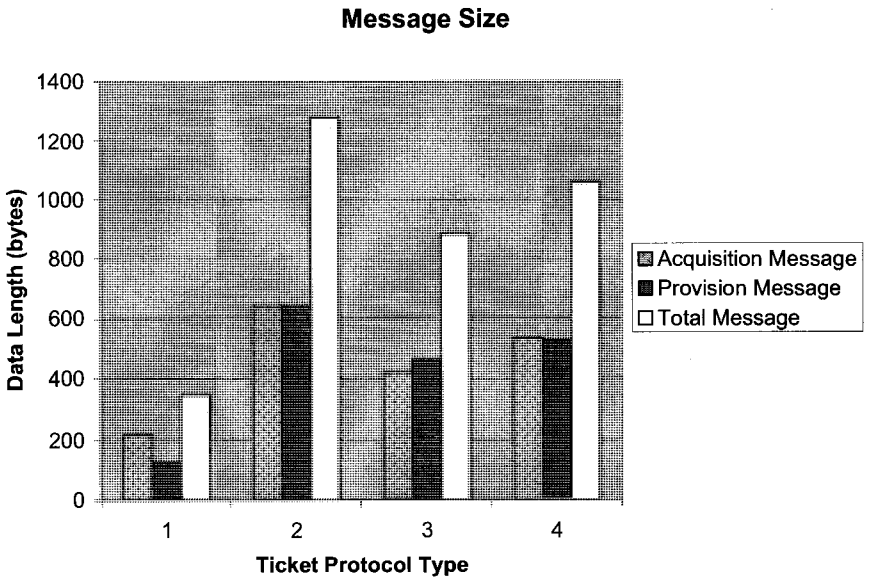

FIGURE 3.2 MESSAGE SIZE FOR ALL PROTOCOLS

ticket acquisition phase while the provision message means all the messages involved during the service provision phase. The total message is just the sum of the acquisition message and the provision message. The total message size used protocol 1 is only nearly 400 bytes, much less than other protocols. This is because other protocols need to send public key or agreement key (1024 bits=128 bytes) in the message. While in protocol 1, the public key is only used for signature generation and verification.

The networks for simulation are two popular packet data service networks: CDMA 1xRTT and GPRS. We implemented the encryption/decryption algorithms with C++ language in a 1.8GHz CPU desktop computer. Thus, the desktop computer acted as the mobile station. Although it is too powerful compared to any mobile station, it is still a good reference as our goal is to compare the performance of protocols.

For in each network, we split the simulation into two phases: ticket acquisition and service provision. For each phase, we simulated the user response time (the time that the user has to spend before replying the message, i.e., verifying and generating data) and the authentication time (the time that the whole protocol must consume from

the first message to the last message) as shown in Figure 3.3 and Figure 3.4:

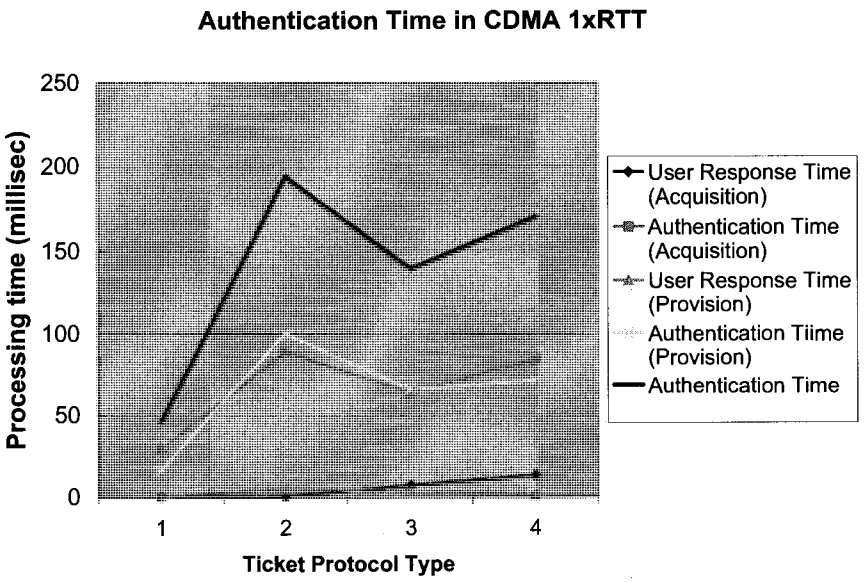

FIGURE 3.3 SIMULATION IN CDMA 1XRTT

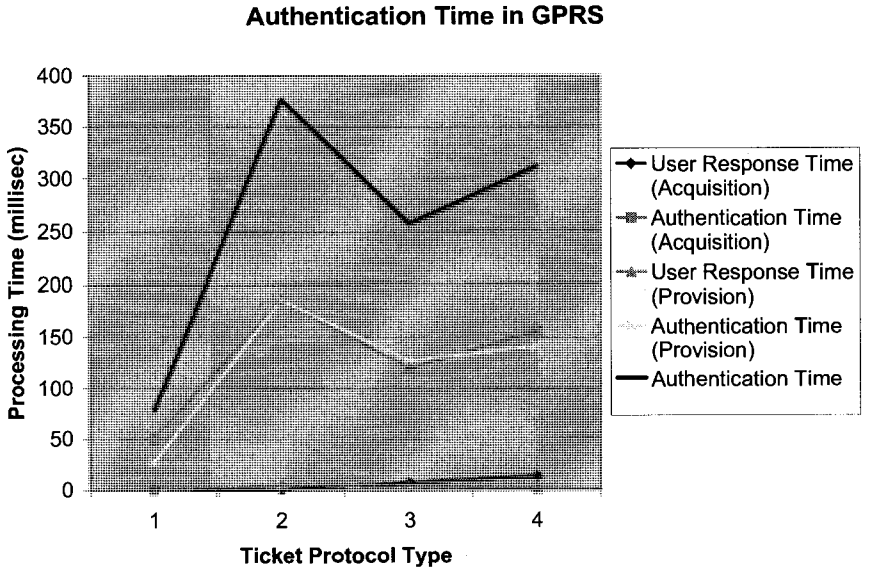

FIGURE 3.4 SIMULATION IN GPRS

The top curve in each figure represents the total authentication time for two phases of the ticket protocols. From simulations in both networks, the user response time for both phases are much less than the service provision time for all ticket protocols.

The user response time of protocol 1 is slightly less than that of other protocols. This difference would be much bigger for real mobile phone. As for the authentication time in both phases, protocol 1 takes much less time than other protocols. This can be explained from the difference of symmetric cryptography and public key cryptography used in the protocols. Considering the transmission time needed in the network, we have the top curves in each figure. These results prove that our proposed mechanism is lightweight.

3.6 Conclusions

Mobile communication systems are bringing more and more choices of VASs to mobile users. Due to various VASPs and numerous mobile users, the guarantee of secure and suitable access is a critical concern in service provision. This thesis proposed a novel mechanism to access mobile VASs through reusable tickets. Considering the limited computation capability of mobile devices, the mechanism is lightweight on the mobile user side: only symmetric encryption/decryption and hash operations are needed. We transfer high computational complexity to the ticket server and the VASP which are usually more powerful. Two hash chains are introduced to achieve mutual authentication, non-repudiation and ticket reuse. By allowing ticket reuse, a mobile user does not have to apply a new ticket for every service access. This is of high interest for two reasons: first, it is convenient for mobile users; second, it decreases the load of the ticket server. In order to avoid reinitializing the payment chain, we introduce a staircase incremental value function associating to the chain. We analyze the protocol through BAN belief logic. Comparisons with classic mechanisms (like Kerberos) and other mobile service access mechanisms, together with the experimental results, show that the proposed mechanism is lightweight, secure and more appropriate to mobile communication.

CHAPTER 4

ENHANCING UMTS AKA WITH VECTOR COMBINATION

4.1 Introduction

The unprecedented growth of world-wide mobile wireless markets, coupled with advances in communication technology and the accelerated development of services taking place in fixed networks, instigates the emergence of third Generation Mobile Communication System (3G). The Universal Mobile Telecommunications System (UMTS) represents an evolution in terms of capacity, data speeds and new service capabilities from second generation mobile networks. It also provides more secure wireless access security mechanisms compared to the Global System for Mobile (GSM) Communications [22].

UMTS wireless security mechanisms has been widely applied while integrating UMTS with other networks [55–62]. Hence, providing secure wireless access mechanisms is critical to wireless network integration. One important mechanism in UMTS is the authentication and key agreement (AKA) [8, 63–65]. UMTS AKA achieves mutual authentication by the user and the network through a long term shared secret key between the user's Universal Subscriber Identity Module (USIM) and the authentication center (AuC) in the user's home environment (HE). The design of UMTS AKA is closely based on the GSM challenge-response scheme. The mobile user's USIM maintains a sequence number with the user's home network. When the mobile user roams to a visited network, the home network sends a set of authentication vectors to the visited network. To authenticate the mobile station, the visited network picks up an unused vector and sends the challenge part to the mobile station. The mobile station checks the freshness of the sequence number and computes a response. Also,

the mobile station computes the cipher key and the integrity key for further communication encryption. The visited network compares the response with the expected response in the authentication vector. If they are same, the mobile station is authenticated. Hence, mutual authentication is realized in UMTS. In GSM, only mobile station is authenticated.

However, UMTS AKA has been criticized due to its introduction of sequence numbers [9–11] and its vulnerabilities of redirection attacks and active attacks in corrupted networks [9]. A failure of sequence number synchronization could result in complicated protocol execution especially when the user roams to a foreign network [8, 10]. The redirection attack can redirect a legitimate user's traffic to an unintended network and also cause billing problem [9]. The active attack in corrupted networks may jeopardize the entire networks because an adversary can use the authentication vectors corrupted from one network to impersonate another network [9].

Another drawback of UMTS AKA exists in the distribution of authentication vectors. A size n array of authentication vectors can be used for n times authentication. Since only the HE is responsible for generating and sending authentication vectors for all subscribers, it actually becomes the traffic bottleneck.

In this chapter, we abandon the adoption of sequence numbers and propose an enhanced AKA to eliminate the above drawbacks. The proposed AKA is based on vector combination: different vectors combining together can also be used for authentication. Through vector combination, a size n array of authentication vectors can realize up to 2^{n-1} times mutual authentication instead of only n times in UMTS AKA. Hence, the traffic for the HE to generate authentication vectors is exponentially decreased.

This chapter is organized as follows. Section 4.2 gives an overview of UMTS AKA. Section 4.3 analyzes briefly its weaknesses. Section 4.4 presents the improved AKA

based on vector combination. Then, security analysis is given in Section 4.4. A comparison with related work is shown in Section 4.6. Finally, we conclude this chapter.

4.2 Overview of UMTS AKA

Figure 4.1 gives an overview of UMTS AKA mechanism. Detailed description can be found in [8,63–65]. The mobile station MS and its HE/HLR share a long term secret key K. This protocol describes the scenario that the MS roams to a visited location register (VLR) or a serving GPRS support node (SGSN). The communication channel between the VLR/SGSN and the HE/HLR is assumed secure [8].

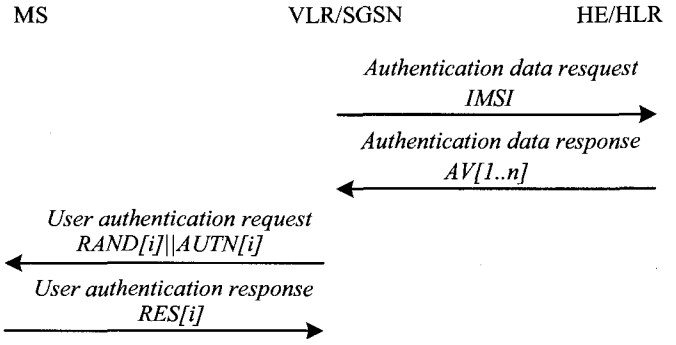

FIGURE 4.1 UMTS AKA

Upon receipt of a request from the VLR/SGSN, the HE/HLR sends an array of ordered authentication vectors $AV[1..n]$ to the VLR/SGSN. The authentication vectors are ordered based on a sequence number. Each authentication vector consists of the following components: a random number $RAND$, an expected response $XRES = f2_K(RAND)$, a cipher key $CK = f3_K(RAND)$, an integrity key $IK = f4_K(RAND)$, an anonymity key $AK = f5_K(RAND)$, a message authentication code $MAC = f1_K(SQN||RAND||AMF)$ and an authentication token $AUTN = SQN \oplus AK||AMF||MAC$, where $f2_K, f3_K, f4_K$ are message authentica-

tion functions with the secret key K, AMF is the authentication management field. The authentication token $AUTN$ includes a sequence number SQN maintained between each pair of HE/HLR and the MS. Each authentication vector can be used for only one authentication and key agreement between the VLR/SGSN and the MS.

Then, the VLR/SGSN selects the next available authentication vector and sends the parameters $RAND$ and $AUTN$ to the user. The MS first checks the correctness of MAC in $AUTN$. Then, it computes the AK and retrieves the $SQN = SQN \oplus AK \oplus AK$. It verifies if SQN is in an acceptable range compared to the value he maintains. If it is not, he rejects the authentication and a sequence number resynchronization procedure may need to be executed by the HE/HLR. Otherwise, after successful verification of $AUTN$, the MS computes a response RES and sends back to the VLR/SGSN. The MS also computes CK and IK as the HE/HLR does. While receiving the response, the VLR/SGSN compares it with $XRES$ stored in the authentication vector. If they match, the VLR/SGSN considers the authentication and key agreement exchange completed successfully.

4.3 UMTS AKA Weaknesses

Although UMTS provides more secure AKA than GSM as claimed by 3GPP [8], there still exists several weaknesses. These weaknesses has been widely discussed in literature. We first give an overview of these weaknesses here and propose an enhanced AKA in the next section.

4.3.1 Sequence number: a bad choice

In UMTS AKA, one of the main drawbacks is the adoption of sequence numbers. The HE/HLR should keep and update a sequence number for every mobile user.

The synchronization and re-synchronization involves complicated work (generation, allocation, verification, and management), especially with regard to the protection against an attack to force the sequence number wrap around and the compromise of user identity confidentiality[10]. A thorough analysis of operational difficulty with sequence numbers is given in [9]. Several scenarios which could result in synchronization failures are discussed in [9, 10]. Once a synchronization fails, the re-synchronization procedure involves quite a few entities and network domains [8]. Moreover, a possible crash in the database which stores the sequence numbers will cost a lot to re-establish the synchronization [9].

4.3.2 HE/HLR: a bottleneck

In UMTS AKA, the HE/HLR is responsible for generating authentication vectors upon receipt of requests from all VLRs/SGSNs. While the number of subscribers is usually large, the HE/HLR experiences heavy authentication traffic and actually becomes the bottleneck of the entire AKA scheme. This is even worse if the mobile user roams to a far away foreign network. This issue has been studied in [66, 67]. Each work proposes an algorithm to investigate the traffic and determine the optimal number of authentication vectors.

4.3.3 Attacks in UMTS AKA

UMTA AKA is proved by BAN logic [29] in [8]. However, it has been suggested that the BAN logic is unable to deal with security flaws resulting from interleaving of protocol steps [47]. Recently, there are two attacks found in [9]: *redirection attacks*, *active attacks in corrupted networks*. The redirection attack occurs when an adversary entices a legitimate mobile station to camp on the radio channels of the false base station. Since the authentication vectors can be used for any serving net-

work, an adversary can intercept the vector and impersonate both the mobile user and the serving network. Consequently, the adversary can redirect user traffic to an unintended network.

The active attack occurs while a network is corrupted and an adversary could forge an authentication data request from the corrupted network to obtain authentication vectors. In addition, the adversary could force the sequence number to be set in a very high value by flooding the authentication data to the home network. As a result, the adversary can start an active attack against legitimate users.

4.4 Vector Combination Based AKA

In this chapter, we propose an enhanced AKA based on vector combination (VC-AKA). In UMTS AKA, each authentication vector is used only once. After that, it should be abandoned in order to defend replay attacks. In fact, we observe that a combination of two vectors can also be used for authentication in condition that this combination should also be used only once. For example, suppose $(RAND_1, XRES_1)$ and $(RAND_2, XRES_2)$ are two challenge-response pairs of two vectors in UMTS AKA, then $(RAND_1 \oplus RAND_2, XRES_1 \oplus XRES_2)$ can also be used for authentication since only the user who passes the single vector authentication can compute the correct response. Of course, extra works need to be done in order to defend replay and impersonation attacks.

In this way, a size n authentication vectors can have up to $(2^n - 1)$ combinations (including combinations with only one vector as in UMTS AKA). The HE/HLR can control the combinations allowed to be used or simply allow both the VLR/SGSN and the MS to use all combinations.

Based on this idea, we propose the protocol two procedures involved as follows. The

first procedure is responsible for the distribution and establishment of authentication vectors; the second procedure realizes vector combination based AKA between the mobile station and the VLR/SGSN (see Figure 4.2).

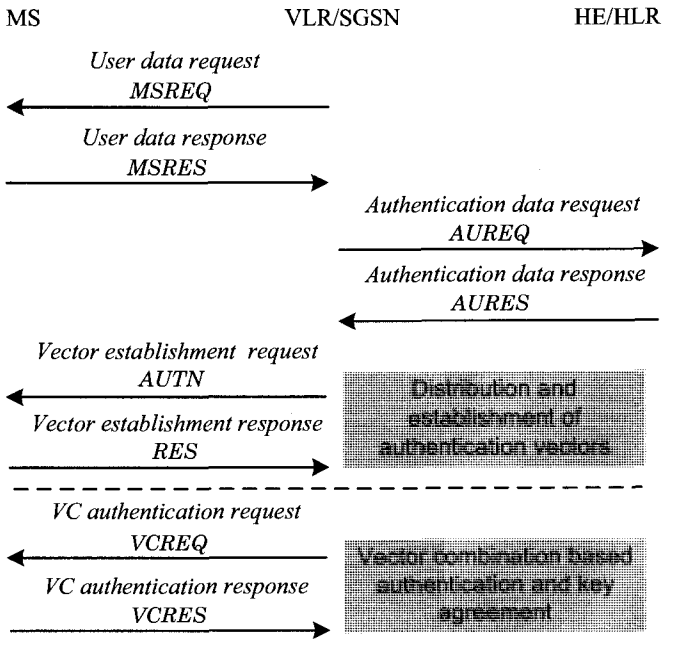

FIGURE 4.2 VECTOR COMBINATION BASED AKA

When a mobile user roams to a visited network for the first time, the VLR/SGSN invokes the first procedure of this protocol by sending $MSREQ$ as *user data request* to the MS:

$$MSREQ = N_V$$

where N_V is a newly generated nonce. Upon receipt of $MSREQ$, the mobile station MS generates $MSRES$ as *user data response* and sends it back to the VLR/SGSN:

$$MSRES = V, H, N_M, MAC_M$$

where N_M is a nonce generated by the MS, and $MAC_M = f1_K(N_M \| N_V \| V \| H)$ is message authentication code generated by the function $f1$ with the secret key

K. The MS indicates that V is the visited network and H is its home network for authentication.

Subsequently, the VLR/SGSN sends $AUREQ$ as *authentication data request* to the HE/HLR:

$$AUREQ = MSRES, IMSI, N_V$$

where $IMSI$ is the MS's International Mobile Subscriber Identity. Upon receipt of $AUREQ$, the HE/HLR verifies if the mobile user is a legitimate user and checks the correctness of the authentication message code MAC_M through the shared secret key K. If it fails, the HE/HLR rejects the authentication request. Otherwise, the HE/HLR succeeds the mobile user authentication. Then, it generates authentication data as follows. First, it generates random numbers R_x, R_y and $RAND$. It computes a response $XRES = f2_K(N_M \| N_V \| RAND)$, a session key $SK = f5_K(N_M \| N_V \| RAND)$ and an authentication token $AUTN = RAND \| N_M \| N_V \| \{R_y\}_K \| MAC$, where $\{R_y\}_K$ means encrypted R_y with the secret key K and $MAC = f1_K(RAND \| N_M \| N_V \| \{R_y\}_K)$. Then, it generates an array of n authentication vectors denoted by $AV[1..n]$. Each vector consists of a challenge-response pair $(RN_i, XRES_i)$, where $RN_i = f5_K(N_M \| N_V \| RAND \| i)$ is the challenge and $XRES_i = f2_K(N_M \| N_V \| RAND \| i) \oplus R_x$ is the response. Both of them rely strongly on the index i. The HE/HLR computes $R = f2_{SK}(R_x \oplus R_y)$ and constructs $AURES$ message as *authentication data response* shown in Figure 4.2:

$$AURES = AV[1..n], R, XRES, SK, AUTN.$$

Upon receipt of $AURES$, the VLR/SGSN sends $AUTN$ as *vector establishment request* to the MS and stores the rest locally. The MS isolates the $RAND, \{R_y\}_K$ and checks the validity of the $AUTN$ by computing the MAC. If it is not valid, the MS rejects. Otherwise, the MS gets the R_y by decrypting $\{R_y\}_K$ with its

secret key K. It computes the session key $SK = f5_K(N_M \| N_V \| RAND)$ and the response $RES = f2_K(N_M \| N_V \| RAND)$. Then, the MS generates an array of vectors $MAV[1..n]$. Each vector $MAV[i]$ consists of a challenge-response pair (XRN_i, RES_i), where $XRN_i = f5_K(N_M \| N_V \| RAND \| i)$ is the challenge and $RES_i = f2_K(N_M \| N_V \| RAND \| i) \oplus R_y$ is the response. It keeps SK and sends RES as *vector establishment response* to the VLR/SGSN.

Upon receipt of the RES, the VLR/SGSN compares if it is the same as $XRES$. If it is not, the VLR/SGSN rejects it and the protocol aborts. Otherwise, the first procedure finishes and authentication vectors are established in both the MS and the VLR/SGSN.

The second procedure is invoked when the VLR/SGSN needs to authenticate the MS. It sends *VC authentication request* to the MS. This request message is based on vector combination. The number of combinations for n vectors is $(2^n - 1)$ (at least one vector in each combination). Each combination, denoted by c, is a 0 and 1 n-bit array and can be used only once. The combination's decimal value is between 1 and $(2^n - 1)$. For example, the decimal value of 01101 is 13 (between 1 and $(2^5 - 1) = 31$). Here, we add a constraint: the number of 1 bits of each combination must be odd. Thus, we actually have 2^{n-1} combinations available.

For each request, the VLR/SGSN chooses randomly a different number c ($1 \leq c \leq (2^n - 1)$) satisfying the above constraint. According to its bit 1 positions $i_1, i_2, \cdots, i_m$, the VLR/SGSN retrieves the challenges $RN_{i_1}, RN_{i_2}, \cdots, RN_{i_m}$ from the vectors $AV[1..n]$. Then, it computes $RN_{VC} = RN_{i_1} \oplus RN_{i_2} \oplus \cdots \oplus RN_{i_m}$ and sends $VCREQ$ as *VC authentication request* to the MS:

$$VCREQ = \{c\}_{SK}, RN_{VC}.$$

Upon receipt of the message, the MS decrypts $\{c\}_{SK}$ and checks if combination c satisfies the constraint and if it is a new combination (all combinations sent from the VLR/SGSN are stored by the MS). If it does not, the MS rejects the authentication request. Otherwise, it accepts the request and retrieves the challenges $XRN_{i_1}, XRN_{i_2}, \cdots, XRN_{i_m}$ according to c from the vectors $MAV[1..n]$ and computes the result $XRN_{VC} = XRN_{i_1} \oplus XRN_{i_2} \oplus \cdots \oplus XRN_{i_m}$.

It checks if RN_{VC} and XRN_{VC} are equal. If they are not, the MS rejects the authentication request. Otherwise, the network authentication succeeds. The MS increases the value of the selected challenges by 1 and stores $XRN_{i_1} + 1, XRN_{i_2} + 1, \cdots, XRN_{i_m} + 1$ into the vector array $MAV[1..n]$ for next time authentication. Then, the MS computes $VCRES$ and sends it as *VC authentication response* to the VLR/SGSN:

$$VCRES = RES_{i_1} \oplus RES_{i_2} \oplus \cdots RES_{i_m} \oplus c.$$

After sending $VCRES$, the MS also generates the cipher key $CK = f3_{SK}(XRN_{VC})$, the integrity key $IK = f4_{SK}(XRN_{VC})$. After that, the MS stores the combination c into its database and updates the number of combinations used. If the number of combinations reaches 2^{n-1}, there is no more combinations available for future authentication. In this case, the MS discards the authentication vectors and the first procedure needs to be executed to request new authentication vectors for future authentication.

Upon receipt of the response, the VLR/SGSN computes

$$VCXRES = XRES_{i_1} \oplus XRES_{i_2} \oplus \cdots XRES_{i_m} \oplus c.$$

Then, it checks if $VCRES$ and $VCXRES$ satisfy the follow equation:

$$f2_{SK}(VCRES \oplus VCXRES) = R.$$

Here, we give an explanation. Since the number of 1 bits in c is odd, $VCRES$ can be written as $A \oplus R_y \oplus c$ and $VCXRES$ can be written as $A \oplus R_x \oplus c$. Thus, $VCRES \oplus VCXRES = R_x \oplus R_y$. Note that $R = f2_{SK}(R_x \oplus R_y)$ is computed in advance by the HE/HLR and stored only at the VLR/SGSN.

If they fail to satisfy this equation, the VLR/SGSN aborts the authentication. Otherwise, the MS authentication succeeds. The VLR/SGSN generates the cipher key and the integrity key similarly as the MS does: the cipher key $CK = f3_{SK}(RN_{VC})$ and the integrity key $IK = f4_{SK}(RN_{VC})$. Similarly, the VLR/SGSN also stores $RN_{i_1} + 1, RN_{i_2} + 1, \cdots, RN_{i_m} + 1$ into the vector array $AV[1..n]$ for next time authentication.

4.5 Security Analysis of VC-AKA

4.5.1 Enhanced Security

The proposed VC-AKA has enhanced security compared to UMTS AKA. In the first procedure, the distribution and establishment of authentication vectors, all the principals (the MS, the VLR/SGSN and the HE/HLR) contribute the generation of the session key SK and each RN_i with their own nonces. While in UMTS AKA, they are generated only by the HE/HLR. In the second procedure, to achieve mutual authentication between the MS and the VLR/SGSN, a randomly chosen combination c is sent to the MS. The response from the MS is generated based on the contributions of all principals. Hence, mutual authentication is stronger. The cipher key CK and the integrity key IK are generated in the similar way.

VC-AKA does not employ sequence numbers to realize network authentication as in UMTS AKA. Instead, all three principals contribute the random numbers' generation and the network authentication is guaranteed through unused vector combinations.

The only extra space needed for the VLR/SGSN and the MS is the storage for used combinations. In order to avoid large space for all 2^{n-1} combinations, the HE/HLR can tune n to a reasonable value (e.g. $n = 7$) or choose randomly only a part of combinations permitted for both the VLR/SGSN and the MS.

If ever $k-1$ vectors are compromised (although it is very hard), the attacker is still not easy to impersonate a legitimate user. For each time authentication, the VLR chooses randomly m vectors. If there is at least one vector not belonging to the subset of $k-1$ vectors, the attacker will not be able to concoct the response. Note that, all the m vectors will be updated after the authentication; thus, the $k-1$ vectors will not be valid any more.

4.5.2 Against Replay Attack

The replay attack is one of main concerns while designing authentication protocols. In UMTS AKA, the sequence number is used to defend against the replay attack. In VC-AKA, we eliminate the sequence number and the VLR/SGSN and the MS maintain a set of combinations. Only if c has not been used before can it be accepted by the MS. Thus, we first defend against the used combinations attack.

As for the authentication vectors, after each time authentication, both the MS and the VLR/SGSN update the selected $RN_{i_1}, RN_{i_2}, \cdots, RN_{i_m}$ by increasing 1. In other words, the challenge RN_i set in the vector $AV[1..n]$ and the challenge XRN_i set in vector $MAV[1..n]$ are updated synchronizely for each time successful authentication. This is required to prevent an attacker concocting a valid RN_{VC} through XOR operations on previously intercepted RN_{VC}s. Thus, we defend against the used challenges attack.

To put the above results together, VC-AKA can defend against the replay attack.

4.5.3 Against Malicious Threats

In UMTS security architecture [8], the VLR/SGSN could be in a foreign network and could be malicious. It is possible that the VLR/SGSN deliberately leaks Alice's information to Eve and tries to help Eve pass the authentication. Suppose that Eve has Alice's $AV[1..n]$. However, since $XRES_i$ is not equal to RES_i, Eve cannot compute the correct $VCRES$ without the knowledge of R_y which is sent to Alice's mobile station secretly.

For each time successful authentication, both the VLR/SGSN and the MS update the challenges by increasing their values by 1. Thus, the challenges in the two arrays are always fresh. Even if an adversary intercepts several used combinations and RN_{VC} values, he can do nothing and cannot concoct a valid RN_{VC} or XRN_{VC} to impersonate a legitimate entity.

The session key SK helps the VLR/SGSN and the MS to generate the cipher key CK and the integrity key IK in the similar way as in [10]. Even if Eve gets Alice's SK in the help of the malicious VLR/SGSN, however, Eve can do nothing. Note that $R = f2_{SK}(R_x \oplus R_y)$ is computed by the HE/HLR in advance; the VLR/SGSN can use R for verification only and cannot obtain the knowledge of R_y due to the one-way hash function; the MS has no knowledge of R and R_x, either.

4.5.4 Against Redirection Attacks

In [9], the authors showed a possible redirection attack in UMTS AKA because the authentication vector can be used by any serving network. In VC-AKA, when the MS roams from the HE/HLR to a VLR/SGSN, it receives the *user data request* message and replies a *user data response* message which includes the $MAC_M = f1_K(N_M \| N_V \| V \| H)$. When the HE/HLR receives the *authentication data request*

message, it can check if the MS is really in the coverage of the supposed VLR/SGSN. If it is not, the HE/HLR refuses the connection request. While the MS receives the authentication token $AUTN$, he can check if they are really sent by the supposed HE/HLR and VLR/SGSN through MAC since all three principals contribute to the generation of MAC. Similarly, for each mutual authentication between the MS and the VLR/SGSN, all principals also contribute to the generation of challenge/response message. Hence, VC-AKA can defend the redirection attack.

4.5.5 Network Corruption Impact

We also examine the impact of network corruption discussed in [9]. Suppose that a VLR/SGSN is corrupted and the adversary can eavesdrop any message received or sent by VLR/SGSN. There are two possible scenarios. First, the MS has no authentication vectors left and needs to execute the first procedure to request vectors from its HE/HLR. However, in this scenario, the adversary cannot forge an authentication data request message because this message should comprise a message authentication code from the MS.

Second, the MS has unused authentication vectors when the VLR/SGSN is corrupted. The adversary can concoct a challenge and force the MS to response. However, the adversary can only use the left combinations and can do nothing when all combinations are used up. In this case, the first procedure should be executed to obtain new authentication vectors while it is impossible as described in the first scenario. Hence, network corruption can only influence the MS who has unused vector combinations. While in UMTS AKA, it could influence all the legitimate users.

4.6 Comparison with Related Work

One of the obvious advantages of VC-AKA is the introduction of vector combination. Consequently, for an array of n vectors, authentication can reach up to 2^{n-1} times instead of only n times in UMTS AKA. Take an example, $n = 7$, and authentication can happen up to 64 times instead of only 7 times. This decreases exponentially the traffic needed to generate authentication vectors for the HE/HLR which has numerous subscribers.

We compare VC-AKA to other proposals. Recently, an analytic model [66] is proposed to investigate the impact of the size of the authentication vector array in order to minimize the cost. Another work [67] proposes a dynamic length of authentication vector array based on prediction of the mobile user's residence time in the VLR/SGSN. Consequently, it is able to reduce the network traffic and avoid the bottleneck at AuC. These works are different from VC-AKA because they do not change the original UMTS AKA protocol and tries to find a best size of the array through traffic analysis.

In [9], an adaptive AKA (AP-AKA) is proposed without sequence numbers. Instead, the MS keeps a list of unused vectors' indexes to verify if an authentication vector is really sent from the serving network and was not used before. This helps the mobile user to defend replay attacks. The protocol also differentiates if the MS stays in his home network or roams to a serving network. In each scenario, the execution of protocol is different. To some extent, this improves the efficiency since the protocol in the home network needs less runs than that in the serving network. As in UMTS AKA, the authentication vectors can also be used only n times.

In [10], the authors employ two techniques: a one-time password/hash chaining technique [12] and keyed-Hash Message Authentication Code (HMAC) [13] instead of the

sequence number to establish mutual authentication. For convenience, we call this HH-AKA. In HH-AKA, when a mobile user roams to a VLR, he generates a hash chain and sends to his HE/HLR. After successful verification of the hash chain, the HE/HLR sends it to the VLR. The VLR then generates another hash chain and sends to the MS. Thus, each MS and VLR pair has two distinct hash chains for successive mutual authentication. This mechanism, however, introduces another synchronization: the authentication times between the MS and the VLR. Once a mismatch of times occurs due to a network failure, it is difficult for the two parties to synchronize. Compared to HH-AKA, the combination check in VC-AKA is more flexible: if a combination is found to be used, it does not influence future authentication. As in UMTS AKA and [9], authentication can happen only up to the length of the hash chain.

As in UMTS AKA, the redirection attack in HH-AKA is also possible as discussed in [9]. An adversary can impersonate a base station and entice the mobile user to camp on the radio channels of the false base station. Since there is no serving network identity indication in the registration message, the home network cannot differentiate whether the message is from the mobile user directly or relayed by an adversary. Consequently, the authentication would be successful and the adversary could redirect the traffic to an unintended network.

Another drawback in HH-AKA is the employment of timestamps for message freshness while the MS sends his hash chain to the HE/HLR. However, to guarantee freshness by timestamps, both MS and HE/HLR must keep local clocks periodically synchronized by a reliable source of time in a secure manner. Between synchronization with the reliable time source, local clocks may drift [14]. Both entities must allow a time window for timestamps to compensate for local clock drift and the fact that messages take time to cross a network. Moreover, the mobile user might have his own "personalized clock" in his real life, for example, always 3 minutes in advance

for not missing the bus. Thus, it is not reasonable to require the mobile user to keep an exact clock as the network.

We simulate the performance for UMTS AKA, AP-AKA, HH-AKA and VC-AKA as they are based on the same model of UMTS AKA. We are particularly interesting in the traffic generated by HLR as it is the bottleneck of the whole performance. Our analysis is based on the data from a week-long trace [68]. A total of 4,156 users place 171,318 packet data calls from 139 cell sites. We assume that each call requires authentication between mobile users and networks. We also assume that the authentication vector length in UMTS AKA is 5 and the length of the authentication vector is 640 bits as defined in 3GPP [69]. The simulation results are shown in Figure 4.3.

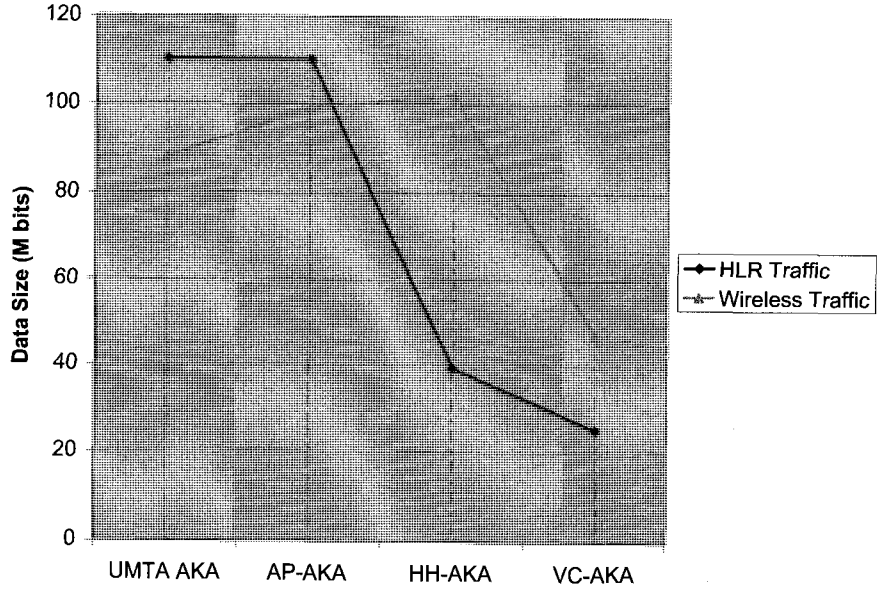

FIGURE 4.3 TRAFFIC COMPARISON FOR AKA VARIANTS

In this figure, the HLR traffic curves represents the authentication traffic that the HLR generates in order to help the VLR/SGSN authenticate the MS. This traffic may include only authentication vectors as in UMTS AKA; or, it may include also other authentication data as in AP-AKA, HH-AKA and VC-AKA. As the authentication vectors in VC-AKA can be used up to $2^{(5-1)} = 16$ times, the HLR needs only generate

171318/16 = 10708 times vector array (5 for each array). While in other AKA variants, the HLR has to generate 171318/5 = 34264 times vector array. From this figure, the HLR traffic for VC-AKA is much less than other variants.

The wireless traffic curves represents the message transferred in wireless radio link: between the VLR/SGSN and the MS. Both the two procedures in each variant may contribute the traffic. In UMTS AKA, there is no wireless traffic involved in the first procedure. While in the second procedure, the VLR/SGSN sends the $RAND_i \| AUTN_i$ to the MS for each time authentication. In AP-AKA, the first procedure contributes some wireless traffic and the second procedure is similar to UMTS AKA. In HH-AKA, the first procedure contributes more traffic than that in AP-AKA and the second procedure sends 2 hash chains for each authentication. In VC-AKA, the first procedure contributes some traffic in order to help the VLR/SGSN and the MS to establish the authentication vectors. While the second procedure is much simpler than other AKA variants. To put them together, the wireless traffic comparison can be seen from this figure. VC-AKA involves least traffic than other variants.

4.7 Conclusions

This chapter first gave an overview of UMTS AKA and showed its weaknesses, especially the adoption of sequence numbers and the traffic bottleneck in the HE/HLR. The synchronization of sequence numbers has been criticized a lot and many improvements have been proposed. In this chapter, in order to eliminate these weaknesses and enhance the security, we abandon completely the adoption sequence numbers and propose an enhanced AKA based on authentication vector combination. The main advantage of VC-AKA is that it liberates the HE/HLR from the bottleneck of authentication vectors generation.

The security analysis showed that VC-AKA can defend against redirection attacks and

active attacks in corrupted networks. Also, VC-AKA provides enhanced security on session key generation and mutual authentication. Through comparison with UMTS AKA and its improvements in literature, we believe that VC-AKA is more efficient and secure.

CHAPTER 5

MOBILITY-ORIENTED AKA IN UMTS

5.1 Introduction

The amazingly high growth of world-wide mobile wireless markets, coupled with advances in communications technology and the accelerated development of services, instigates the emergence of third Generation Mobile Communications System (3G). The Universal Mobile Telecommunications System (UMTS), a standard of 3G, represents an evolution in terms of capacity and more secure service capabilities from second generation mobile networks.

In our real life, although people have the potential for relatively random patterns of behavior, there are easily identifiable routines in every person's life [70]: getting out of bed, eating lunch, driving from home to office, weekend shopping, etc. In mobile networks, not surprisingly, most users follow also regular routine during business hours, residing mostly at their place of work [71]. Many users show little or no mobility and they generally visit a small portion of the network [68, 72]. An understanding of user mobility is required for the design, the implementation and the proper provisioning of services in 3G and future cellular networks [68], [72], [73].

In Chapter 4, we analyzed UMTS AKA and discussed its weaknesses. In this chapter, we consider authentication and user privacy issues in UMTS from the point of view of the user mobility patterns. The UMTS AKA does not consider the user mobility pattern. It treats the mobility for all mobile users in a general way: the same authentication through a challenge-response mechanism. The challenge-response mechanism involves always three entities: the mobile user, the home environment/home location

register (HE/HLR) and the visitor location register (VLR). Consequently, they are not efficient if mobile users stay in their frequently visited foreign networks for most of their activities.

To improve the AKA mechanisms to be mobility-oriented, we propose *MO-AKA* in which each mobile user keeps a cache of the most frequently visited VLRs and the last visited VLR. Each VLR in the cache can be a *proxy* of the HE/HLR for authentication purpose. Upon the next authentication, the mobile user first checks the cache to see if there is a proper proxy available. If he can find one, the mobile user and the newly visited VLR do not need to run long distance protocol with the HE/HLR for authentication.

This chapter is organized as follows. Section 5.2 gives the background and the motivation. Our proposed mobility-oriented authentication is presented in Section 5.3. Then, we analyze our approach in Section 5.4. A comparison and performance is given in Section 5.5. Finally, we concludes this chapter.

5.2 Background and Motivation

In [72], the authors analyzed a seven-week trace of a metropolitan-area wireless network in San Francisco Bay Area. The results shows that 42% of all mobile users are stationary, and that 64% visit only one location per day. More surpringly, as the number of locations visited on a daily basis increases, the average distance traveled decrease. This implies that the more mobile users roam, the closer the locations are.

From the empirical findings, Halepovic *et al.* [68] introduced a notion of *home cell* in their mobility model. The notion of home cell refers to the cell from which a user makes a significant proportions of calls. They found that 55% of the users are stationary and 9.6% had only a single location change during the week.

In UMTS, a VLR usually comprises many cells and AKA is realized at the VLR level, not the cell level. Hence, different locations can be in the same VLR and most activities of a single mobile user could be in one VLR. We call this VLR as *home VLR*. Compared to the mobility behavior in terms of location, we can obtain the following observation.

Observation: *In UMTS, practically, a single mobile user usually has a fairly regular routine and could have a* home VLR *in which occur his most activities. Such mobile users have a high proportion of all mobile users.*

Quintero [71] indicates if mobile users are classified into categories, the system could treat each category differently to minimize system costs. This is helpful to traffic routing, admission control, resource allocation and handoff management [74], [75]. Such understanding is critical for planning future large-scale mobile network infrastructure such as UMTS [68], [72], [73].

Naturally, mobility patterns should be taken into account for the design of network security. Based on our observation, we classify users into two categories: *stationary users* and *mobile users*. Stationary users are those who stay in a VLR for most network service requests. Mobile users are those who roam frequently among VLRs. In UMTS AKA, we notice that the HE/HLR is a bottleneck of authentication traffic. If we could find a way to treat stationary users and mobile users differently and decrease the whole traffic, we would eliminate the bottleneck finally.

5.3 Mobility-Oriented Authentication

For convenience, we denote the identity of the mobile user by M, the identity of VLR/SGSN by V and the identity of HE/HLR by H. We assume that a mobile user shares a secret key K_{MH} with the HE/HLR. In addition, we assume that the HE, the

VLR/SGSN have their own secret key K_H, K_V respectively. For the first time that a mobile user registers to the HE/HLR, an encrypted identity $\{IMSI\}_{K_H}$ is stored in the MS. The MS can present this encrypted identity for authentication request.

5.3.1 The Trust Model

In UMTS AKA, the trust relationship exists in two ways: one between the MS and the HE/HLR, the other between the HE/HLR and other VLRs. That's the reason why the VLR/SGSN should fetch an authentication vector generated by the HE/HLR. Also, it is the reason why the HE/HLR is the traffic bottleneck. Based on our observation, we may establish a "trust relationship" between the MS and his home VLR. Of course, this trust relationship should be under the control of the HE/HLR. In other words, only the HE/HLR can help both the MS and the home VLR trust each other. This would be reasonable in if the HE/HLR and the VLR belong to the same wireless operator. Formally, we have the following assumption:

Assumption: *In UMTS, the HE/HLR may delegate its authentication right to the MS's home VLR.*

Note that this delegation, and the trust relationship, should be short-term and the HE/HLR should update it periodically. The trust model is shown in Figure 5.1.

In this model, the solid line represents the system predefined trust relationship. The empty line between the MS and the home VLR indicates an established (through the HE/HLR) trust relationship. The HE/HLR registers the home VLR of the user (usually the stationary user) based on his mobility behavior. This registration could be done through an agreement between the user and the network or through machine learning.

In UMTS AKA, when a VLR/SGSN succeeds the authentication of a mobile user

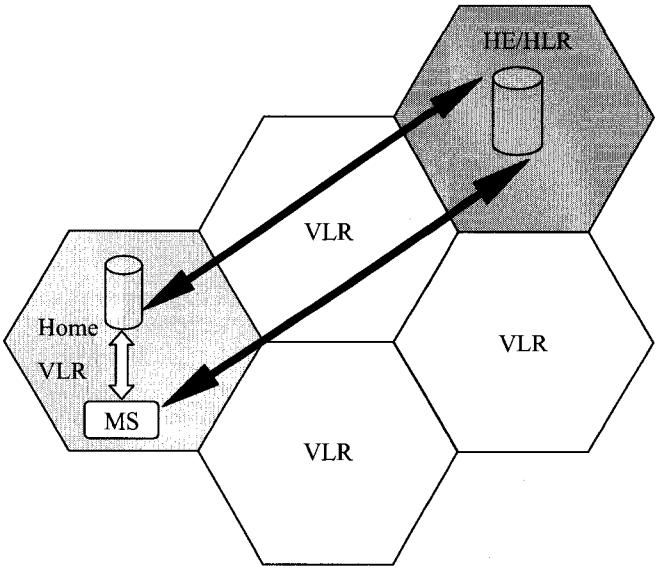

FIGURE 5.1 THE TRUST MODEL

with the help of his HE/HLR, a mutual trust relationship is established between the VLR/SGSN and the MS. This means that this VLR/SGSN could be helpful if the MS roams to another VLR/SGSN. To maintain the relationship, we need a *shared secret* between the VLR/SGSN and the MS. When the MS starts to roam for the first time, the HE/HLR helps the VLR/SGSN and the MS to establish a shared secret key. When the MS roams to another VLR/SGSN, he can ask his last visited VLR/SGSN to be responsible to establish a trust relationship between him and the new VLR/SGSN. In this case, the visited VLR/SGSN is a kind of *proxy* of the HE/HLR and there is no necessity to run long distance authentication with the HE/HLR.

5.3.2 Authentication Process

Based on our observation, to be efficient and practical, we propose that the MS keeps a cache of VLRs/SGSNs which are the most frequently visited VLRs/SGSNs by the MS (for example, at home and at the office). Certainly, this cache includes his home VLR. Besides the home VLR in the cache, we allow a few other VLRs whose

activities with the MS have also relatively high proportion. According to the results from [68], [72], [76], the number of VLRs in the cache could range from 1 to 5. These VLRs/SGSNs can be learned step by step based on some user profile strategies [71], [75], [77], [78], [79], [80] or simply based on a user profile definition in practice.

Since the newly visited VLR/SGSN is usually near to the last visited VLR/SGSN according to [72], we add a special property to the cache. Besides the most frequently visited VLRs/SGSNs, the first entry of the cache is always the last visited VLR/SGSN. During the service provisioning of the entire network, once the mobile user wants a VLR to be in his cache, he notices the VLR to keep the shared secret key for fairly long time. Periodically, each VLR will check the visit frequency of all such mobile users in its database and delete those users who visit much less or do not visit any more.

When the MS roams to a VLR/SGSN, he first checks if it is in the cache. If so, he authenticates with the VLR/SGSN directly without involvement of a third party. Otherwise, he asks the last visited VLR/SGSN as a proxy to authenticate himself. After that, he updates the cache by replacing the first entry as the newly visited VLR/SGSN. Through this way, we add intelligence into UMTS AKA and call this *MO-AKA*. In practice, the user mobility pattern definition or learning should be updated periodically in order to improve the hit rate of the cache. Nevertheless, we should notice that the VLRs/SGSNs are not completely equal to the HE/HLR. When the MS leaves his HE/HLR and roams for a long time, it is necessary to re-authenticate the MS through the HE/HLR.

In the following, we will consider different scenarios: *MS roams from the HE/HLR to a VLR/SGSN*, *MS Roams between VLRs/SGSNs*, *MS Revisits a VLR/SGSN* and *MS Revisits the HE/HLR*.

5.3.3 MS Roams from the HE/HLR to a VLR/SGSN

The first scenario is the most discussed one in literature: the MS leaves his HE/HLR and starts to roam to a VLR/SGSN. If it is the first time, the cache is empty. In this case, the only way through which the VLR/SGSN authenticates the mobile user is with the help of the HE/HLR. The figure 5.2 shows the protocol runs in this scenario.

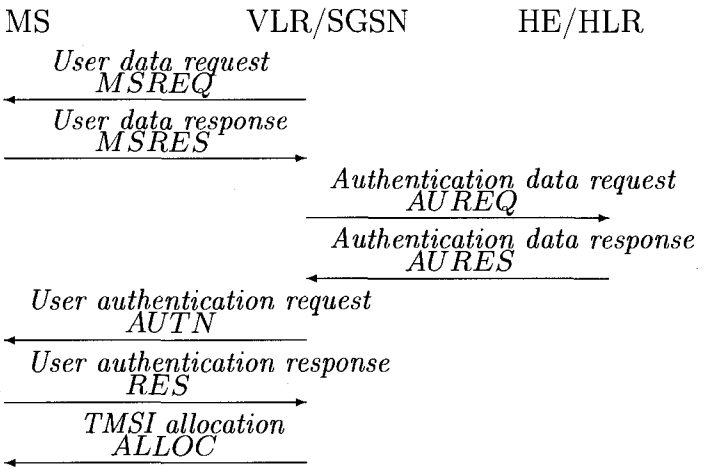

FIGURE 5.2 MS ROAMS FROM HE/HLR TO VLR/SGSN

When the mobile user roams to the visited network, the VLR/SGSN invokes the protocol by sending $MSREQ$ message for user data request. The $MSREQ$ includes a newly generated nonce N_V. Upon receipt of $MSREQ$ message, the mobile station generates $MSRES$ message: $MSRES = \{IMSI\}_{K_H}, H, N_M, MAC_M$ where N_M is a nonce generated by the mobile user, and $MAC_M = f1_{K_{MH}}(N_M \| N_V \| V \| H)$ is message authentication code generated by the function $f1$ with the secret key K_{MH}. The MS indicates that H is his home network for authentication. Subsequently, the VLR/SGSN sends $AUREQ$ for authentication request to the HE/HLR. The $AUREQ$ consists of $MSRES$ and N_V: $AUREQ = MSRES, N_V, V$.

Upon receipt of $AUREQ$, the HE/HLR first decrypts $\{IMSI\}_{K_H}$ to get the identity of the mobile user and retrieves the shared secret key K_{MH}. After the verification

that the mobile user is a valid registered user, the HE/HLR checks the correctness of the authentication message code MAC_M through the shared secret key K_{MH}. If it fails, the HE/HLR rejects the authentication procedure. Otherwise, the HE/HLR succeeds the mobile user authentication. Then, it generates a random number $RAND$ and computes a shared key K_{MV} between the mobile user and VLR/SGSN: $K_{MV} = f3_{K_{MH}}(N_M \| N_V \| RAND)$. This shared key is much secure since all the three parties contribute their own nonces. The HE/HLR includes this secret key in the $AURES$ message: $AURES = RAND, AMF, IMSI, MAC_H, K_{MV}$, where AMF is authentication management field and $MAC_H = f1_{K_{MH}}(RAND \| AMF \| IMSI \| V)$. It sends this message back to the VLR/SGSN.

Upon receipt of $AURES$, the VLR/SGSN keeps the shared key K_{MV} and builds an authentication token $AUTN$ message: $AUTN = RAND, AMF, MAC_H, MAC_V$, where $MAC_V = f1_{K_{MV}}(RAND \| AMF \| MAC_H \| N_V)$, and sends to the mobile user. Upon receipt of $AUTN$, the mobile user first verifies MAC_H through the shared key K_{MH}. If it is not correct, he aborts the authentication procedure. Otherwise, the mobile user succeeds the authentication of the HE/HLR. Then, he computes the secret key $K_{MV} = f3_{K_{MH}}(N_M \| N_V \| RAND)$ and continues to verify the correctness of MAC_V with this key. If it is correct, he succeeds the authentication of the VLR/SGSN. Then, he computes the cipher key $CK = f3_{K_{MV}}(N_M \| N_V \| RAND)$ and integrity key $IK = f4_{K_{MV}}(N_M \| N_V \| RAND)$ for further secure communication. Also, he computes RES and sends back to the VLR/SGSN: $RES = f2_{K_{MV}}(N_M \| N_V \| RAND), \{RN\}_{K_{MV}}$ where RN is a new random number generated by the MS.

The VLR/SGSN verifies the correctness of RES through the newly established shared key K_{MV}. If it fails, the VLR/SGSN rejects the authentication. Otherwise, the VLR/SGSN succeeds the authentication of the mobile user. Then, the VLR/SGSN generates a $TMSI$ and stores the triplet $(IMSI, TMSI, K_{MV})$ in its local database

for future authentication. In the database, the $IMSI$ is the primary key and $TMSI$ should be uniquely indexed. This means that there should be only one entry in the database for each $IMSI$ and the triplet can be retrieved by a $TMSI$. If there is already an entry for an $IMSI$, it should be updated.

The VLR/SGSN also computes the cipher key: $CK = f3_{K_{MV}}(N_M \| N_V \| RAND)$ and the integrity key: $IK = f4_{K_{MV}}(N_M \| N_V \| RAND)$ for subsequent secure communication. At last, the VLR/SGSN generates $ALLOC$ message and sends to the mobile user: $ALLOC = \{\{TMSI\}_{K_V}, RN + 1, TMSI\}_{K_{MV}}$ where $TMSI$ is the newly generated temporary mobile subscriber identity. It should be noticed that $TMSI$ is usually coupled with location area identity (LAI) [8] in order to locate the mobile user's position in the network. For simplicity, we only put $TMSI$ in the protocol. Upon receipt the $ALLOC$, the mobile user verifies its correctness by decrypting it with K_{MV} and checking the nonce $RN + 1$. If it is correct, he saves $\{TMSI\}_{K_V}$ and $TMSI$ for next time authentication use.

Until now, the mobile user authenticates the VLR/SGSN and the HE/HLR; the VLR/SGSN and the HE/HLR succeed the authentication of the mobile user. Thus, mutual authentication is realized. The MS adds this VLR/SGSN and the K_{MV} into his cache if it is considered to be a frequently visited VLR/SGSN.

5.3.4 MS Roams between VLRs/SGSNs

The second scenario occurs when the mobile user roams from a VLR/SGSN to another VLR/SGSN and the latter is not in the cache of the mobile user. The new VLR/SGSN does not need to ask the HE/HLR for authentication. Instead, the last visited VLR/SGSN (denoted by VLR_o/SGSN_o) would be a proxy of the HE/HLR to authenticate the mobile user. The authentication procedure is similar to the one that the MS roams from his HE/HLR to a VLR/SGSN. We just need to replace the

HE/HLR identity H with V_o. The only difference is that the user temporary identity $\{TMSI\}_{K_{V_o}}$ is sent instead of the permanent identity $\{IMSI\}_{K_H}$ in the $MSRES$ message. We just omit the detailed description here.

5.3.5 MS Revisits a VLR/SGSN

The next scenario of MO-AKA considers the authentication while the mobile user revisits or remains in a VLR/SGSN. According to our observation, this scenario is the most common in practice. Most of activities of a given mobile user are found in this scenario. Thus, the efficiency of AKA in this case is critical to the overall performance of wireless networks. The authentication procedure is very simple as shown in figure 5.3.

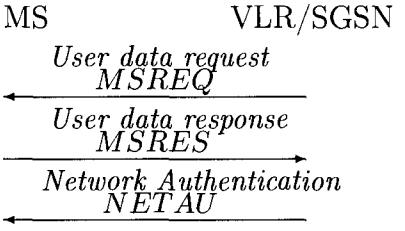

FIGURE 5.3 MS REVISITS A VLR/SGSN

The VLR/SGSN invokes the protocol by sending $MSREQ$ message for user data request. The $MSREQ$ includes a newly generated nonce N_V as in other scenarios. Upon receipt of $MSREQ$, the mobile station generates $MSRES$ message $MSRES = \{TMSI\}_{K_V}, \{N_V, V, N_M\}_{K_{MV}}$ and sends back to VLR/SGSN, where N_M is a nonce generated by the mobile user. When the VLR/SGSN receives the message, it decrypts the first field with its secret key K_V and gets the mobile user's $TMSI$. It retrieves the triplet from its database and gets the corresponding shared key K_{MV}. Then, it decrypts the second field with this key and gets the nonce N_V, V and N_M. It checks the correctness of the nonce N_V and the identity V. If they are correct, the

VLR/SGSN authenticates the mobile user and knows that the mobile user knows V as his authentication network. It computes the cipher key $CK = f3_{K_{MV}}(N_M \| N_V)$ and the integrity key $IK = f4_{K_{MV}}(N_M \| N_V)$ for subsequent secure communication. Then, it generates a new $TMSI$ encrypted with its secret key for the mobile user and sends back $NETAU$ message: $NETAU = \{N_M \oplus IMSI, \{TMSI_n\}_{K_V}\}_{K_{MV}}$ where $IMSI$ is retrieved from the triplet.

As soon as the mobile user receives the $NETAU$ message and verifies the correctness, he succeeds the authentication of the VLR/SGSN. He stores $\{TMSI_n\}_{K_V}$ for next time authentication. Then, he computes the cipher key $CK = f3_{K_{MV}}(N_M \| N_V)$ and the integrity key $IK = f4_{K_{MV}}(N_M \| N_V)$ for subsequent secure communication. Thus, mutual authentication between the mobile user and the VLR/SGSN is realized.

5.3.6 MS Revisits the HE/HLR

Now, we consider the last scenario of MO-AKA: the mobile user revisits his HE/HLR. This could be the case that the mobile user roams from a VLR/SGSN or the mobile user stays in the HE/HLR and just reauthenticates. In this scenario, the mobile user does not need to ask a VLR/SGSN for authentication as in the previous scenario. The authentication is also simple as shown in figure 5.4.

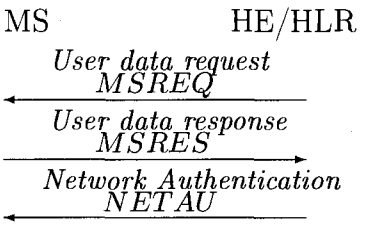

FIGURE 5.4 MS REVISITS THE HE/HLR

The HE/HLR invokes the protocol by sending $MSREQ$ message for user data request. The $MSREQ$ includes a newly generated nonce N_H. Upon receipt of $MSREQ$, the

mobile user generates $MSRES$ message and sends back to HE/HLR: $MSRES = \{IMSI\}_{K_H}, \{N_H, H, N_M\}_{K_{MH}}$ where N_M is a nonce generated by the mobile user. When the HE/HLR receives the message, it decrypts the first field with its secret key K_H and gets the mobile user's $IMSI$. It retrieves the corresponding shared key K_{MH}. Then, it decrypts the second field with this key and gets the nonce N_H, H and N_M. It checks the correctness of the nonce N_H and the identity H. If they are correct, the HE/HLR authenticates the mobile user and knows that the mobile user knows H as his home network. Then, it computes the cipher key $CK = f3_{K_{MH}}(N_M \| N_H)$ and the integrity key $IK = f4_{K_{MH}}(N_M \| N_H)$ for subsequent secure communication. It sends back $NETAU$ message: $NETAU = \{N_M \oplus IMSI\}_{K_{MH}}$.

As soon as the mobile user receives the $NETAU$ message and verifies the correctness, he succeeds the authentication of the VLR/SGSN. Then, he computes the cipher key $CK = f3_{K_{MH}}(N_M \| N_H)$ and the integrity key $IK = f4_{K_{MH}}(N_M \| N_H)$ for subsequent secure communication. Thus, mutual authentication between the mobile user and the HE/HLR is realized.

5.4 Security Analysis of MO-AKA

5.4.1 User Privacy and Untraceability

In mobile networks, *untraceability* is the ability that it is difficult to verify the history, location, or application of an user by means of recorded identification. Untraceability belongs to user privacy category and is recommended by 3GPP. In all the protocols of MO-AKA, the user identity is always sent in an encrypted form. To prevent an adversary from tracing the encrypted identification, a $TMSI$ is generated and encrypted by the VLR once the MS the VLR establish a trust relationship with the help of the HE/HLR. After that, the $TMSI$ is changed immediately whenever the

MS re-visits the VLR. While in UMTS AKA, the $IMSI$ is still sent in cleartext while the mobile user sends registration requests to the VLR. Hence, we realize user privacy and untraceability.

5.4.2 Key Generation and Freshness

In the first two scenarios, the shared secret key between a VLR and a MS is generated by the HE/HLR or the last visited VLR in this way: the key is the function of three nonces from the VLR, the MS and the HE/HLR (or the last visited VLR). The cipher key and the integrity key are also generated in the same way in the first two scenarios. For the last two scenarios, two parties are involved instead of three parties. The cipher key and the integrity key are generated by the function of two nonces from the MS and the VLR (or the HE/HLR). Hence, the generated secure keys are strong and fresh since all involved parties contribute their nonces.

5.4.3 Threats and Attacks

MO-AKA is a tailored UMTS AKA which takes into account the user's mobility pattern. It is based on an assumption and works only when the user's home VLR is determined and available. The assumption is the base stone of MO-AKA. The agreement is made by the MS, the home VLR and the HE/HLR. Any principal can terminate the agreement and MO-AKA will be replaced by regular UMTS AKA. Also, the agreement needs to deal with the issue that the shared secret between the MS and the home VLR should be updated periodically.

This assumption would be easy within the same wireless operator where the HE/HLR and all VLRs can be trusted each other. Between different wireless operators, however, the management of the trust relationship is critical in MO-AKA and may bring

risks from both the inside and the outside of the system. The home VLR and the VLR in cache from a different operator might be malicious and deliberately leak the secret key K_{MV} to other mobile users to help them impersonate the legitimate users. Consequently, the VLR may blame that the legitimate user leaks the secret key and it is hard to justify the responsibility.

The management of the cache is also a risky task to the system. The updating of the cache may bring potential outside attacks, for example, DoS attacks. An adversary may launch numerous updating requests to a VLR. The VLR, as a result, becomes very busy in maintaining the database and may not be able to handle regular service requests. In order to defend the DoS attacks, we can avoid the cache updating procedure. In order to avoid updating, in practice, the cache size can be fixed and limited to 1 (home VLR only) or 2 (home VLR + a most frequently visited VLR). This would be also beneficial for the management of the trust relationship. In addition, it would be fairly easier to designate a trustful home VLR than require other VLRs be trustful as well. Thus, the probability of threats from inside would be much less.

5.5 Comparison and Performance Evaluation

MO-AKA treats stationary users and mobile users differently. As a result, it does not need sequence number synchronization any more and decreases the network traffic and improve the authentication efficiency. Considering the mobility behavior of mobile users in practice, MO-AKA is a tailored AKA which improves the efficiency and liberates the HE/HLR from the burden of authentication vectors requests without loss of security. We decrease the authentication traffic not only in the HE/HLR part, but also in the overall network because authentication between a VLR and a mobile user is lightweight while the VLR is in the cache of the mobile user.

We evaluate the performance of MO-AKA. Our analysis is based on the data from

a week-long trace [68]. A total of 4,156 users place 171,318 packet data calls from 139 cell sites. We assume that each call requires authentication between mobile users and networks. We suppose the number of authentication vectors in UMTS AKA is 5 and the length of the authentication vector is 640 bits [69]. We define that there are 5 VLRs/SGSNs in the cache of MO-AKA. Among the 5 VLRs/SGSNs, the first one is always the last visited VLR/SGSN and others are the most frequently visited VLRs/SGSNs (for example, at home and at the office).

For simplicity, we also suppose that there is no synchronization needed of sequence numbers in UMTS AKA. All networks work normally without corruption. We simulate different AKA mechanisms and calculate the network traffic generated in this one week-long trace. We define the stationary traffic as the authentication traffic involved by stationary users and the mobile traffic as the authentication traffic involved by mobile users. The total traffic is simply the sum of all users (including stationary and mobile users). Figure 5.5 compares total traffic, stationary traffic and mobile traffic in these four AKA mechanisms.

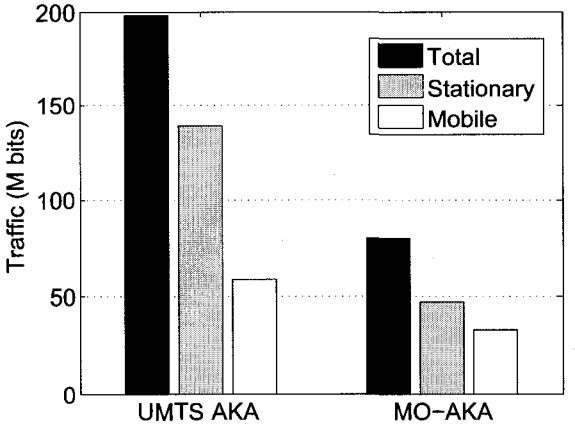

FIGURE 5.5 TOTAL TRAFFIC, STATIONARY TRAFFIC AND MOBILE TRAFFIC

From this figure, we can see that the total traffic in both UMTS AKA is much higher that in MO-AKA. This is determined by the fact that all authentication vectors are generated by the HE/HLR and transfered to VLRs, while this is not the case

in MO-AKA. In UMTS AKA, the HE/HLR is responsible to generate a batch of authentication vectors. While in MO-AKA, the HE/HLR generates only a temporary key for the VLR and the MS. And the following authentication will not require the authentication vectors any more before an explicit temporary key update required. Thus, it transfers a part of authentication task to the VLR.

As for the stationary traffic, UMTS AKA generates much more than MO-AKA. This is because there is no involvement of the HE/HLR while a mobile user remains in a VLR. All the authentication will be processed with this security key and no authentication vectors needed. Further, the authentication message in this case is shorter than that in UMTS AKA.

As for the mobile traffic, the traffic difference is not so obvious as in stationary case. For mobile users roaming frequently among different VLR/SGSNs, the authentication is performed through standard UMTS AKA. In this case, the generated authentication traffic will be the same as in UMTS AKA. MO-AKA is based on the mobility pattern of users, and would not improve the performance for mobile users.

Nevertheless, MO-AKA generates slightly less traffic. This is because MO-AKA keeps a cache of VLRs and does not need to ask HE/HLR for authentication if the VLR can be found in the cache.

We can also compare the traffic involved at the wireless link and at the HE/HLR in figure 5.6. We define the wireless link as the traffic involved between the MS and the VLR and the HE/HLR traffic as the necessary traffic generated by the HE/HLR in order to generate the authentication vectors. MO-AKA requires much less HE/HLR traffic than UMTS AKA as the HE/HLR does not have to generate the authentication vectors for all authentications. This illustrates well that the established trust relationship in MO-AKA liberates the HE/HLR from the traffic bottleneck in UMTS AKA.

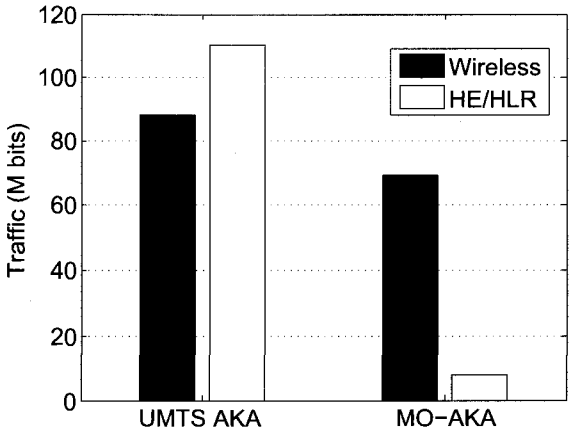

FIGURE 5.6 WIRELESS TRAFFIC AND HE/HLR TRAFFIC

The proxy VLR in the cache can also help the network shorten the response time for authentication requests because the newly visited VLR does not need to run long distance protocol with the HE/HLR. Thus, it improves the network performance efficiency. Further detailed analysis will require the knowledge of the network topology. Different topologies should be considered and analyzed in order to evaluate our approach. We do not discuss here since our focus in this chapter is the security aspect.

5.6 Conclusions

In this chapter, we first analyzed the security weaknesses in UMTS AKA. Specifically, we showed the traffic bottleneck at HE/HLR in AKA as well as the drawbacks of sequence numbers. Then, based on the empirical results on mobile user behaviors, we find that mobile users have fairly regular routines in the daily life. The mobility pattern is the motivation of our proposed mobility-oriented AKA.

The basic idea of MO-AKA is to treat stationary users and mobile users differently. To reach the goal, we introduce a trust relationship established between a VLR and a mobile user. This relationship can not only simplify the authentication procedure between two entities, but also act a proxy of the HE/HLR to help another VLR

to establish the trust relationship with the mobile user. A security analysis shows that MO-AKA provides enhanced user privacy and authentication security. Also, a cache of the most frequently visited VLRs is introduced in order to eliminate unnecessary frequent long distance run protocol with the HE/HLR. The benefit from these mechanisms is remarkable traffic decrease, particularly the traffic involved at the HE/HLR which is usually the bottleneck in UMTS AKA.

Through security analysis and the performance comparison on a week-long trace in practice, we show that our approach is more efficient and secure. For future work, we may analysis other performance indicators, such as the network response time, given the topology of a practical mobile network.

CHAPTER 6

A FINE-GRAINED PUZZLE AGAINST DOS ATTACKS

6.1 Introduction

The convergence of mobile communication and Internet brings Internet-like mobile services and specific services like mobile commerce directly to mobile users. The UMTS system architecture and protocols are designed to accommodate these services, while they expose the UMTS network to various Internet attacks as well. The Denial-of-Service (DoS) attack is one of these attacks. The UMTS public land mobile network (PLMN) can be easily congested and therefore paralyzed by bogus traffic from the Internet. These attacks happen not only in "pull" type services invoked by user equipments, but also in "push" type services invoked by the network node.

In addition, the increasingly powerful mobile devices can coordinate to launch typical DoS attacks, taking advantage of limited resources in the network, such as the radio link bandwidth. These devices include Palm OS [81], Symbian OS [82] and Smartphone OS [83]. With open APIs and powerful services, these devices can access Internet and wireless mobile networks like UMTS, and potentially launch DoS attacks.

The function of a DoS attack is fundamentally to flood its target servers with numerous connect requests to prevent them from being accessible to any other legitimate users' requests or providing services. Consequently, the target servers become too overburdened to respond to requests from its attackers. As a result, it has insufficient system resources to respond to legitimate requests on the network. The CSI/FBI 2004 Computer Crime and Security Survey found that other than viruses, DoS attacks caused more financial losses than any other cyber-security breach.

In UMTS networks, the danger of DoS attacks is threatening different targets [1]: the communication infrastructure, specific services and the mobile devices. Given the high risk of DoS attacks for UMTS networks, it is vital to provide a systematic countermeasure to thwart these attacks. In addition, the system needs a mechanism to control the access of legitimate users according to current resource consummation.

There are a number of defenses against DoS attacks in literature. Meadows [84] proposed a formal framework based on an observation that any point during a protocol execution where a responder may accept a bogus message as genuine could be used to launch a DoS attack. The framework defines cost functions which assign costs of engaging in individual actions. Then, it compares the cost to defender against the cost to attacker. A protocol is called DoS resistant if the former is less than the latter.

Another defense to prevent DoS attacks is client puzzle technique [16, 18, 20, 85]. Client puzzles have been used to provide service guarantees to legitimate clients. For each service request, the client is forced to solve a cryptographic *puzzle* before the server commits its resources. The computational efforts for each request is very small. However, this imposes a large computational task on adversaries generating traffic in large quantities. The main idea behind client puzzles is that any client requesting service must allocate some of its own resources (processing time or memory) before the server commits its resources for the connection. The client must consume resources to solve the puzzle, while the puzzles are usually verified easily by the servers.

These client puzzles, however, are usually designed for wired network environment. Most of them are not small enough to accommodate wireless access networks. In addition, the difficulties of these puzzles are not easily to be adjusted finely, which results in a gap between two neighbor difficulty levels.

In this chapter, we propose a *quasi partial collision* concept and design a new client puzzle based on it. Our puzzle is very small; the difficulties can be fine-grained

controlled. In Section II, a discussion of related work is given. Then, we propose our quasi partial collision technique in Section III. We analyze it and design the client puzzle in Section IV. A comparison and performance results are shown in Section V. At last, we conclude this chapter.

6.2 Related Work

Aura [15] proposed that a good puzzle should have certain properties. For convenience, we list these properties plus two more properties specially for mobile networks.

1. Creating a puzzle and verifying the solution are not expensive for the server;
2. The cost of solving the puzzle is easy to adjust from zero to impossible;
3. The puzzle can be solved on most types of client hardware;
4. It is not possible to precompute solutions to the puzzles;
5. While the client is solving the puzzle, the server does not need to store the solution or other client-specific data;
6. The same puzzle may be given to several clients; Knowing the solution of one or more clients does not help a new client in solving the puzzle;
7. A client can reuse a puzzle by creating several instances of it;
8. The difficulty of a puzzle can be fine-grained controlled by the server;
9. The puzzle should be small and compact in order to avoid too much increase in traffic.

Compared to the second property, the eighth property is added in order to emphasize that the server can adjust finely the difficulties of puzzles instead of bringing a gap

while increasing or decreasing a difficulty level. This is particularly useful for access control. The last property is required in mobile networks where radio bandwidth is limited compared in wired networks.

There are several puzzles mechanisms. Time-lock puzzles require a client to spend a particular amount of computation time performing repeated squaring [19] before communication. These puzzles require the calculation of two large prime numbers in the server side. It consumes significantly the system resources. Thus, these puzzles cannot be generated efficiently on the order of microseconds.

Another puzzle mechanism is to force clients to reverse one-way hashes calculated at the server given the original random input with n bits erased [17]. These puzzles are small and compact, and the puzzle generation time is significantly smaller than time-lock puzzles. In order to vary the difficulty level, n is either increased or decreased. The client performs a brute-force search on the erased bits by hashing each pattern in the space until it finds the answer. The big disadvantage of these puzzles is that the effort of solving a $(k+1)$ bits puzzle is twice as hard as that of a k bit puzzle. In [17], another approach based on multiple hash-reversals is proposed to mitigate the difficulty. However, this brings more traffic workload and is not efficient in UMTS environments. Feng *et al.* [16] proposed an improved version through a hint of the solution. The server sends the puzzle along with a hint of the answer that gives the client an idea of where the answer lies. The hint is a single value. By adjusting the hint, the difficulty of a client puzzle can be probabilistically controlled in a range. However, in this approach, the server should run the hash function to get a hint before sending it to the client. When the client sends back the solution, the server have to run the hash function again or compare the solution with the one stored in database. This needs more system resource consumption.

Aura *et al.* [15] showed a technique based on partial collision in a hash function. A good hash function (like SHA1, MD5) requires that no two distinct strings whose

hash values are same. A partial collision is that two hashed values have partially identical strings. In this puzzle protocol (Figure 6.1), the server generates nonce N_S and the difficulty level k to the client.

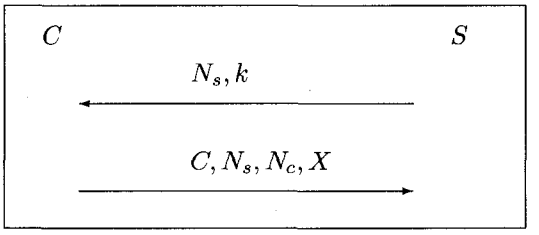

FIGURE 6.1 PUZZLES BASED ON PARTIAL COLLISIONS

The client generates his own nonce N_C. The necessity of N_c is to prevent an attacker to solve the solution based on N_s and send to the server before the client does. The client spends its time to find a required partial collision and from the following equation and sends the solution (C, N_s, N_c, X) to the server.

$$h(C, N_s, N_c, X) = \overbrace{000\cdots00}^{k \text{ bits 0's}} \underbrace{x_1 x_2 \cdots}_{\text{other bits}}$$

h	=	a hash function (MD5 or SHA1)
C	=	the client identity
N_s	=	the server's nonce
N_c	=	the client's nonce
X	=	the solution of the puzzle
k	=	the difficulty level of the puzzle
$000\cdots00$	=	the k first bits, must all be 0
$x_1 x_2 \cdots$	=	other bits, garbage data

In this equation, the client should find a string whose hashed value has partial collision (k bits) with given hashed values. In Aura's approach, the hashed value of the solved

puzzle must be k-bit (the most significant bits) partially collided with all 0's string. The difficulty level is determined by k, the number of collision bits. If $k = 0$, no work is required. If $k = 128$ (for MD5) or $k = 160$ (for SHA1), the client must reverse the entire one-way hash function, which is computationally impossible.

However, Aura's approach has a drawback: the effort of solving a $(k+1)$-bit puzzle is twice as hard as that of solving a k-bit puzzle. This does not meet the eighth puzzle property. Table 6.1 shows some examples tested by a Java application in Windows Desktop with Intel CPU 350MHz.

TABLE 6.1 AVERAGE TIME NEEDED TO FIND PARTIAL COLLISIONS

No.	Collision Bits	Time (seconds)
1	17	2
2	18	4
3	19	7
4	20	13
5	21	25
6	22	47
7	23	103
8	24	4*60
9	25	8*60
10	26	15*60
11	27	32*60

From Figure 6.2, the time needed to find partial collisions grows exponentially while the number of bits increases.

Thus, there is a big gap between the two nearest difficulties. We need a method to finely control the difficulties quite smoothly in order to have a better performance.

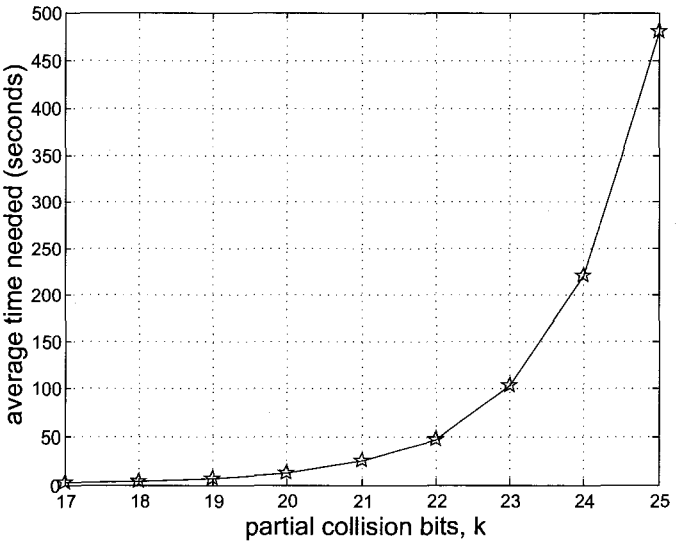

FIGURE 6.2 AVERAGE TIME NEEDED TO FIND PARTIAL COLLISIONS

6.3 Quasi Partial Collision

We reconsider the idea of partial collision puzzles. Given a k-bit puzzle, the client is required to find a solution whose hash value cares only the first k-bit values and does not care the $(k+1)$-bit (x_1) value and other bits values.

In the $(k+1)$-bit puzzle, the x_1 should be considered and its value must also be zero while it can be 0 or 1 as in the k-bit puzzle. Thus, more work should be done to find a new solution whose hash value has the most significant $(k+1)$ zero bits.

$$h(C, N_s, N_c, X) = \overbrace{000\cdots00}^{k \text{ bits } 0\text{'s}}\mathbf{0}\ \underbrace{x_2x_3\cdots}_{\text{other bits}}$$

Thus, the need of one extra bit zero is a strong condition. We ought to make a weak condition in order to compensate the whole requirement. Since this is only one bit, we cannot make use of it too much. We might ask its neighbor bits for help.

Our proposed puzzle protocol is shown in Figure 6.3. While applying this protocol

to UMTS environment, the server can be service providers or operators who are the potential DoS targets.

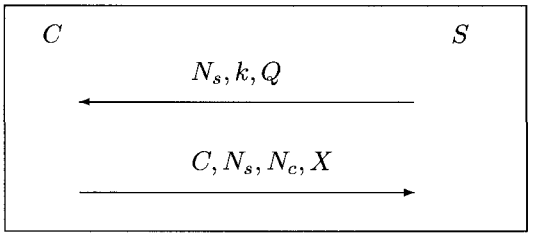

FIGURE 6.3 PUZZLES BASED ON QUASI PARTIAL COLLISIONS

This is similar to Aura's approach except that the server also decides a *quasi value* Q. The quasi value is used to control the quasi partial collisions. We will explain value later. These three values together form the puzzle that is sent to the client.

To solve the puzzle, the client also generates a random nonce N_c. Now, we modify a little the equation, for example, by considering the last 4 bits $Q_1Q_2Q_3Q_4$ in the first k-bits as the following.

$$h(C, N_s, N_c, X) = \overbrace{000\cdots00}^{(k-4)\ \mathbf{0's}} \underbrace{Q_1Q_2Q_3Q_4}_{\mathbf{4\ quasi\ bits}} \overbrace{x_1x_2\cdots}^{\mathbf{other\ bits}}$$

This equation is similar to Aura's equation. In this example, however, we have an extra requirement. If we isolate the 4 bits $Q_1Q_2Q_3Q_4$, its decimal value must be less than Q.

$$Q_1Q_2Q_3Q_4 < Q$$

Here, we also use k to control the difficulty level. Since there is a big gap between $(k-1)$-bit partial collision puzzle and k-bit partial collision puzzle, we allow the last four bits, $Q_1Q_2Q_3Q_4$, of the first k bits to take several acceptable values instead of only 0000 in a normal k-bit partial collision puzzle. And we still treat the whole k bits as collision. Thus, we call this *quasi* partial collision (Figure 6.4). In other words, we

require a *weak* k-bit partial collision.

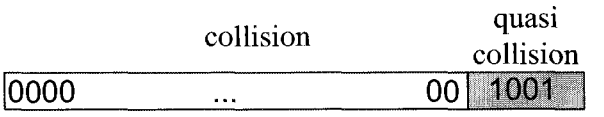

FIGURE 6.4 k BITS QUASI COLLISION

For a good hash function, we assume that the hash values of a set of random input messages distribute randomly. There are 16 combinations for these four bits from 0000 to 1111. We define a set $\mathbb{A}$ with Q elements chosen from these 16 combinations. Based on this assumption, the probability of a hash value whose $Q_1Q_2Q_3Q_4$ is in the set $\mathbb{A}$ is dependant only on Q and independent on the elements in $\mathbb{A}$. Hence, for efficiency and simplicity, we choose the smallest Q strings and require that $Q_1Q_2Q_3Q_4 < Q$. The value of Q is called *quasi value*. The quasi value Q from the server controls the number of choices. We can change the decimal value of the four quasi bits between 0 and 15. If $Q = 1$, then this equation becomes a normal k-bit partial collision puzzles. If $Q = 16$, then it is a $(k-4)$-bit partial collision puzzle. Any other values will generate a puzzle between k-bit and $(k-4)$-bit puzzles. In this way, we add several difficulty levels into the big gap. The selection of Q is practical, so we need experiences to determine proper values.

In short, quasi partial collision puzzles ask the client to give solutions whose hash values are not strictly collided but in an allowable value range. Thus, we decrease the difficulty.

6.4 Fine-grained Control over Difficulties

Our goal is to design a client puzzle which meets the nine required properties. It is required that the difficulty of a puzzle could be fine-grained controlled. We just discussed an example of quasi partial collision. Now, let's step back and think this

problem again.

The difficulty grows exponentially while the number of bits needed for partial collision increases. In fact, the last bit (as each other bit) in $(k+1)$-bit partial collision is forced to be 0. This constraint makes its difficulty twice as hard as k-bit partial collision because in the latter case the $(k+1)$ bit can be 0 or 1. Hence, if we denote D_k the difficulty of k-bit partial collision, then we have

$$D_k = 2 * D_{(k-1)} = 2^2 * D_{(k-2)} = 2^3 * D_{(k-3)} = \cdots \tag{6.1}$$

We look at this problem from a different view. If we allow the last bit of $(k+1)$-bit partial collision to have two choices (0 and 1), then this problem becomes k-bit partial collision. In other words, if we allow two $(k+1)$-bit strings acceptable as solutions of $(k+1)$-bit partial collision puzzles, the difficulty of the puzzles decreases to be half. Thus, if we can find a way to give more acceptable strings, then we can have finer scale of difficulties.

Let us look at again the example in the previous section. There are 16 choices for 4 bits. If we fix only one choice for these bits whatever it is 0000 or 1010 or some other combinations, the difficulty level is D_k. If we allow two choices for these 4 bits, then it is equivalent to $(k-1)$-bit partial collision since it is essentially same as we ignore the last bit value. We denote the new difficulty as $D_{k,2}$ (2 means two acceptable strings, or quasi value), and we have $D_{k,2} = D_k/2$. Similarly, we have $D_{k,8} = D_k/3$, $D_{k,16} = D_k/4$. Now, we ignore the number of quasi bits and only care the number of acceptable strings (in fact, we can work in a reversal way, from the latter to get the former). Suppose we allow Q acceptable strings in k-bit partial collision puzzles, the client must first decide how many last bits (in the k first bits) needed, for example, 3 bits needed for $Q = 6$, 4 bits needed for $Q = 10$. These bits are quasi bits. Then, the client should find a solution that the decimal value of quasi bits in the hashed

value is not larger than Q. The following is the final version of the equation in our approach.

$$h(C, N_s, k, Q, N_c, X) = \overbrace{000\cdots00}^{(k-m)\ \textbf{0's}} \underbrace{Q_1 Q_2 \cdots Q_m}_{m\ \textbf{quasi bits}} \overbrace{x_1 x_2 \cdots}^{\textbf{other bits}}$$

If we denote the difficulty for a k bit puzzle with Q quasi value as $D_{k,Q}$, then we have the following equation,

$$D_{k,Q} = D_k/Q \tag{6.2}$$

From equations 6.1 and 6.2, it is easy to see that

$$\begin{aligned}
D_k &= D_{k,1} \\
D_k &= 2 * D_{(k-1)} \\
D_{k,2} &= D_{(k-1)} \\
D_{k,3} &= D_k/3 \\
D_{k,4} &= D_{(k-2)} \\
D_{(k-2)} &< D_k/3 < D_{(k-1)} \\
D_{(k-3)} &= D_{k,8} < D_{k,7} < \cdots < D_{k,4} = D_{(k-2)} \\
D_{(k-4)} &= D_{k,16} < D_{k,15} < \cdots < D_{k,8} = D_{(k-3)} \\
\vdots &\quad \vdots
\end{aligned}$$

In fact, the normal partial collision is a special case of quasi partial collision with the quasi value $Q = 1$. By introducing the quasi value, we add new difficulty levels between two neighbor levels like $D_{(k-3)}$ and $D_{(k-2)}$ in Aura's approach. The gap between these two levels becomes much smaller than before. Thus, we have a fine-grained scale of puzzle difficulties.

Based on the above analysis, we notice that the additional difficulty levels introduced

by quasi partial collisions become closer and denser while the quasi value increases. This give us a guide on how to choose k and Q in practice. For example, if we want to add about 6 difficulty levels between $k = 20$ and $k = 21$ in normal partial collisions, we choose $k' = 23$ (because $(2^2) = 4 < 6 < 8 = (2^3)$) and set Q from 4 to 8. Thus, we obtain

$$D_{20} = D_{23,8} < D_{23,7} < \cdots < D_{23,4} = D_{21}$$

For general cases, this is the rule to choose k' and Q. First, we strengthen the difficulty by requiring several more bits collisions; then, we weaken the difficulty by allowing appropriate quasi values. The more additional levels we need, the larger are the k' and the Q.

6.5 Comparison and Performance Discussion

We analyze our approach by comparing the properties defined in Section II. First, the puzzle can be verified quickly (a hash function) by the server. Second, we can adjust the difficulty level k according to current system performance. Moreover, our approach provides a fine-grained control over difficulties. This meets the eighth property.

As indicated in [15], client puzzle works only in an authentication protocol to defend DoS attacks. The server sends a puzzle to the client before it allocats its resource. For flooding attacks, the clients or attackers will not try to resolve the puzzle. Thus, client puzzle will not be able to defend the flooding attacks in this case.

In an authentication protocol, when the client is solving the puzzle, the server does not have to store the solution. The server needs only to check the nonces and justify if it is a new puzzle. Hence, the system resource consumption is limited.

Compared to the hint-based hash-reversal puzzles [16], the amount of time to solve a quasi partial collision puzzle is deterministic, not probabilistic as in the hint approach. In addition, in the hint approach, to generate a puzzle with $O(D)$ difficulty, the server should generate a hint which needs to be uniformly distributed in $(0, 2D)$ difficulty range. In our approach, it is much easier since the server needs only to send a quasi value Q.

Compared to the multiple hash-reversal puzzles, our puzzles are smaller in the size. It meets the last property. It is practical and efficient in UMTS systems. Moreover, it is easy to control the difficulties instead of combining different sizes of puzzles into a new puzzle.

Since the quasi partial collision is similar to partial collision, our puzzles meet other properties as Aura's approach.

We simulate the time needed to find quasi partial collisions for $k = 21, 22, 24, 25$ and the quasi value ranging from 1 to 11 (Figure 6.5). In order to verify whether we reach the goal for fine-grained controlling over difficulties, we compare the number of difficulty levels in both partial collisions and quasi partial collisions. For simplicity and clarity, we consider only the difficulty levels from 0 and 100 seconds (Figure 6.6).

In this figure, the first column represents the difficulty levels in partial collision while others represent the difficulty levels in quasi partial collisions with different k bits. Each quasi partial collision introduces new levels in different ranges. Compared to the difficulty level distribution in the partial collision, all the new levels provide much finer granularity. For example, in the partial collision, there is only one level in the range 20-30 seconds; in the quasi collisions, there are about 8 levels. Note that we can choose more k and q values to obtain more levels.

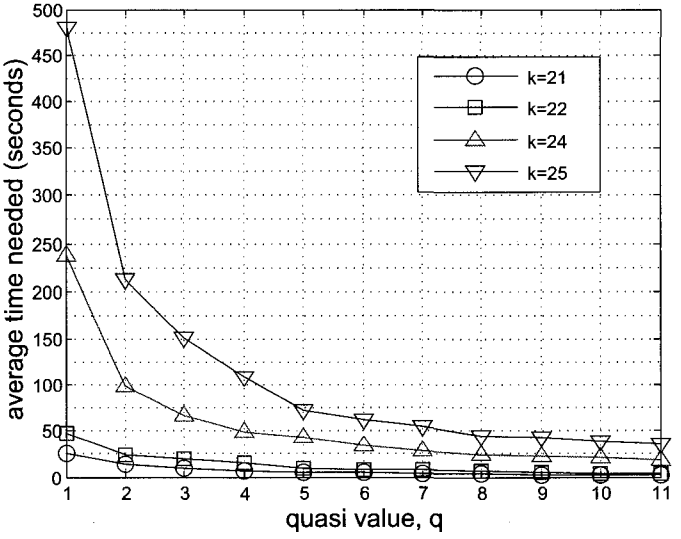

FIGURE 6.5 AVERAGE TIME NEEDED TO FIND QUASI PARTIAL COLLISIONS

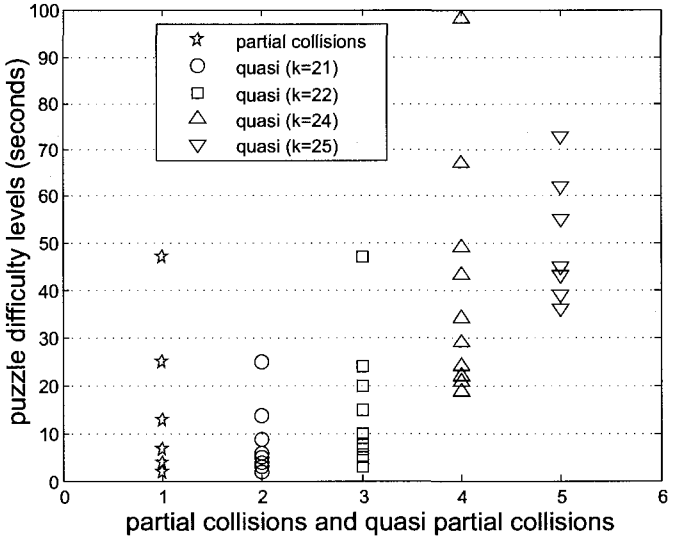

FIGURE 6.6 PUZZLE DIFFICULTY LEVELS COMPARISON

6.6 Conclusions

In this chapter, we show that UMTS is vulnerable to DoS attacks while it brings Internet-like services to mobile users. We described a client puzzle approach based on quasi partial collisions to defend these attacks. Actually, this mechanism can also be applied in other networks which are vulnerable of DoS attacks.

Currently, most types of client puzzles have a common drawback: solving a higher difficulty level puzzle is about twice as hard as the effort to solve the current one. By introducing quasi partial collisions, we add more difficulty levels between the gap. This is realized by adjusting the quasi value. The quasi value allows that some last bits of the partial collision do not have to be 0 as required in the normal partial collision approach. Instead, they can have some choices and we still treat them as collisions. Thus, we have fine-grained control over difficulty puzzles. This is particularly useful for access control. We analyzed and examined our approach. In practice, the k and Q should be carefully chosen based on the client computational power and the current system performance in order to obtain expected effects.

CHAPTER 7

GENERAL DISCUSSION

In the first chapter, we discussed that mobile devices are generally characterized by the limited memory capacity and limited computational power in mobile networks. Also, due to the mobility, the mobile user and the service provider should be mutually authenticated while the user roaming to a foreign domain. Moreover, the wireless communication medium is more difficult to be protected against active attacks and eavesdropping. An adversary can easily intercept the messages sent through the radio link. These constraints have prevented a simple migration of cryptographic protocols that have been widely adopted in wire-line networks to wireless networks for authentication and security. Based on these constraints, we proposed our research objectives.

In this chapter, we discuss our designed mechanisms with these constraints. First, we analyze the lightweight techniques used for mobile devices. Second, we analyze our mechanisms and claim that they can prevent replay attacks and provide confidentiality, integrity and user privacy. Then, we discuss the performance of our designed mechanisms and compare to related works. Finally, we investigate the conformity of our contribution to the proposed research objectives.

7.1 Lightweight to Mobile Devices

Due to limited computational complexity of mobile devices, lightweight techniques such as hash function and hash function chain are adopted in our designed mechanisms. Hash function is much faster than public key cryptography operations. In the

proposed ticket-based service access mechanism, we apply a hash chain into the ticket instead of Diffie-Hellman key agreement as in other related work. In the proposed client puzzle against DoS attacks in UMTS, a hash function is also used to build a client puzzle.

However, the application of hash function in these two mechanisms are quite different. In the first mechanism, the hash function is used for non-repudiation and authentication. This is guaranteed due to the one-way property of a hash function. In the second mechanism, the hash function is used for finding partial collisions.

Other lightweight techniques can also be found in UMTS AKA enhancements. In both VC-AKA and MO-AKA, only symmetric key cryptographic operations, key generation functions (i.e., $f2_K(x), f5_K(x)$) and MAC functions are used as bricks in the protocol. These operations are generally much faster than public key cryptographic operations.

7.2 Against Replay Attacks

As an adversary can easily intercept a message sent during the protocol runs, replay attacks are one of the concerns in security protocol design. In ticket-based service access mechanism, we adopt a hash chain in order to prevent replay attacks. Both the mobile user and the service provider maintain a shared hash chain. For each time of service request, the mobile user takes off the next available hash node from the hash chain and presents to the service provider. The latter validates the request by verifying the correctness of the hash node with its current hash chain. After successful authentication, both them update the hash chain. Even if an adversary intercepts the message, it cannot replay it as a valid request since the service provider will not accept it.

In VC-AKA, both the VLR/SGSN and the MS keep updating the authentication arrays for every time authentication request. Once some challenges in the arrays have been used for current authentication request, they will be updated immediately if the authentication finishes successfully. An adversary cannot rebuild a valid request based on the challenges intercepted during previous authentication requests. In MO-AKA, a newly generated TMSI and a nonce are sent by the MS in order to prevent replay attacks.

7.3 Confidentiality, Integrity and User Privacy

Another concern in mobile networks is the data confidentiality while the messages transferring through wireless link. In ticket-based service access mechanism, all the messages are encrypted either by the ticket server and the service provider's public key, or by the mobile user's long term secret key and the fresh session key. Thus, an eavesdropper can learn nothing from the protocol runs. In VC-AKA, the eavesdropper can get the random numbers in cleartext. However, he cannot do anything due to the integrity guaranteed by the MAC. During the authentication in the second procedure, the eavesdropper can only get the encrypted combinations and corresponding challenges. Nothing more information can be obtained based on these values. Similarly in MO-AKA, all messages are encrypted by the shared secret key between the VLR/SGSN and the mobile user.

User privacy is also a critical issue in mobile communications. As we introduced previously, one of the advantages of tickets is the user privacy. A ticket does not include user personal information while the user requests a service. Wherever the user roams, he will not leak his personal information to the service provider. While in VC-AKA and MO-AKA, they are based on UMTS AKA which protects user privacy.

7.4 Performance Discussion

7.4.1 Ticket-based Service Access

When a mobile user needs to acquire a ticket through a VASP, it is mandatory that the ticket server be accessible. This is similar to Kerberos [21] in which the server should always remain on-line. However, the situation of our proposal is less serious because this protocol would not happen frequently. It can be mitigated by using multiple ticket servers. It seems that the system executes cross domain authentication and loses the benefit of a ticket. However, since the ticket described in this paper is reusable, one time cross domain authentication is worth future reused local domain authentication.

The authentication chain allows a ticket to be reused $N \times T$ times. Nevertheless, ticket reuse does not mean a duplicate ticket is acceptable because it would fail in authentication without a correct hash node. In order to use this chain correctly, both the user and the VASP have to synchronize its current position in the chain. If the synchronization fails (e.g., the user loses the position information), it is difficult for them to resynchronize. Consequently, the ticket should be abandoned. As for the payment chain, if it is not well designed, it can run out before the service terminates. Experiments are helpful to determine the length of the chain and the value function.

Since the ticket server is so critical for secure service access, it may experience denial-of-service (DoS) attacks. Meadows [84] shows that any point during a protocol execution where a responder may accept a bogus message as genuine could be used to launch a DoS attack. The client puzzle technique [16, 18, 20, 85–87] is widely used against DoS attacks and to provide service guarantees to legitimate clients. For each service request, the client is forced to solve a cryptographic puzzle before the server commits its resources. The computational complexity for each request is very small.

However, this imposes a large computational task on attackers generating a large quantity of traffic and slows down the requests significantly. Since mobile devices are usually low in the power, an efficient and finely-controlled DoS mechanism should be employed. The proposed client puzzle in Chapter 6 can be used for this purpose.

7.4.2 Enhancements of UMTS AKA

The proposed VC-AKA has two main advantages. First, there is no need for synchronization of sequence numbers between the MS and the HE/HLR. Second, the HE/HLR is no longer the bottleneck of authentication traffic generation.

In VC-AKA, there are three runs of message exchange for the first procedure. It seems that VC-AKA is not so efficient as UMTS AKA (two runs only). This can be explained from two aspects. First, the purpose of the first run in VC-AKA is to establish the knowledge that both the MS and the VLR/SGSN know each other as its designated principal. Also, both of them contribute its nonce for future use. Thus, security is enhanced. Second, one of the purposes of the first procedure is to establish two different authentication vector arrays (size n) for the MS and the VLR/SGSN. These arrays can be used for future authentication up to 2^{n-1} times. Hence, one extra run is worth the efficiency of entire AKA.

7.4.3 Client Puzzles Against DoS Attacks

The proposed client puzzle against DoS attacks in UMTS can also be applied in other mobile networks, such as UMTS, even Internet. The hash function can also be replaced by others such as MD5 and SHA1. An encryption function is also possible to replace the hash function.

In every environment, the quasi value and the difficulty level need to chosen based

on experiments. Also, the fine-grained difficulty control is based on an assumption: the hash values of a set of random input messages distribute randomly. In practice, however, this assumption may not be always true. Consequently, solving some puzzles can take longer or shorter time than expected. This requires also the necessity of experiments.

7.5 The Relations of Mechanisms in Networks

In mobile communications, the mobile user communicates with the server via wireless link. Wireless link brings a whole new meaning to design network security mechanisms. There is an ongoing debate on this issue: Which layer should we protect? Link layer, network layer or even higher layer (transport or application layer)? Here, we discuss our mechanisms with network layers.

Link layer security protects a wireless network by denying access to the network itself before a user is successfully authenticated. This prevents attacks against the network infrastructure and protects the network from attacks that rely on having IP connectivity in network layer. However, link layer security protocols do not provide end-to-end security. On the other hand, network layer or transport layer can provide more suitable security solution in wired portion of the networks. Thus, it is generally accepted that link layer security can be complemented through the use of security mechanisms at higher protocol layers. Link layer security protocols may be used to provide network access domain security, including user authentication to the serving network. High layer security protocols can provide end-to-end security.

The designed mechanisms are particularly applicable to next-generation mobile networks where a number of mobile systems coexist in a large-scale manner. To put them into the network model, the enhancements of UMTS AKA authenticate mobile users at the link layer while the ticket-based service access is responsible to provide the

service in network layer, transport layer or even in application layer. As a result, the service provision will be as follows. The mobile user first needs to get authenticated to the mobile networks through the UMTS link layer authentication mechanisms; then, he can further send a service provision request to his local service provider through the ticket-based service access mechanism. This would involve large-scale authentication. The mechanism can provide flexibility and scalability in this scenario.

Without a doubt, the service providers and UMTS home networks could be the potential targets of DoS attacks during the service provision. The designed client puzzles can be launched to defend against these attacks if the system finds it necessary.

In short, our designed mechanisms provide suitable authentication solutions for next-generation mobile network not only from the link layer but also from the network layer or even high layer. We also give a fine-grained solution for potential DoS attacks.

7.6 Contribution Integration into UMTS networks

In this thesis, we proposed four contributions, each of which focuses on different aspects of the network. Considering the next-generation mobile networks, for example, UMTS networks, these contributions can be integrated into UMTS and improve the overall performence of the network.

Mobile service access through ticket is an upper layer mechanism. We need to designate a CA to be trusted by all service providers like Bell mobility, Verizon wireless, etc. We also need to establish a ticket server which is able to responde mobile user's requests to issue tickets. This ticket server should be trusted by the mobile users and all service providers. Thus, a user can obtain the ticket and use the ticket for service request. In this scenario, the user does not need to sign a contract with each service provider in order to get the service.

UMTS AKA improvements can be used for data link layer improvement. VC-AKA is specially useful for multiple service providers. However, the mobile user needs to maintain its authentication vectors with different service provider. This is the extra cost compared to the standard UMTS AKA. If there is only one carrier in a country, MO-AKA can be used to improve the overall performance.

7.7 Conformity of the Contribution to the Research Objectives

In this section, we compare our designed mechanisms to the research objectives proposed in the beginning of this thesis.

The first specific research objective is to propose an authentication mechanism for mobile value-added service access which takes into account the constraints in mobile networks and shifts high computational efforts from limited power mobile devices to powerful servers. In our work, the designed ticket-based mobile service access realizes mutual authentication between the mobile devices and the service provider. It requires mobile devices only lightweight operations: hash function and symmetric cryptographic operations. Expensive computational operations such as public cryptographic operations are only employed in the powerful ticket server and service providers. Considering the constraints in mobile networks, this mechanism can prevent replay attacks and provide confidentiality, integrity and user privacy. As the proposed ticket is reusable and the payment is based on a staircase incremental value hash chain, it is more efficient than other related mobile service access mechanisms. Thus, the designed mechanism realized the proposed specific research objective.

The second specific research objective is to propose enhancements of UMTS authentication and key agreement mechanism. We proposed two enhancements in this thesis. The original UMTS AKA was designed specifically for mobile networks with the consideration of the constraints. Our enhancements are based on the original UMTS

AKA without introducing extra expensive operations. Both enhancements improve the efficiency by liberating the HE/HLR from the authentication traffic. To reach this goal, one enhancement (VC-AKA) is based on vector combination while the other (MO-AKA) is based on user mobility pattern. In VC-AKA, a set of authentication vectors can be reused securely many times, thus it reduces authentication vector traffic needed from the HE/HLR. In MO-AKA, the home VLR can help the HE/HLR authenticate the mobile user. In addition, both provides enhanced security compared to the original UMTS AKA. Thus, we realized the proposed specific research objective.

The third specific research objective is to design a client puzzle mechanism against DoS attacks in UMTS. The designed client puzzle is based on a lightweight hash function. The client need to find a solution to meet the quasi partial collision requirements. The difficulty level can be tuned easily in order to provide fine-grained control compared to related works on client puzzles. It is compact and suitable to mobile devices. Considering the characteristics of UMTS PLMN networks, the designed client puzzle is appropriate to difficulty control and effective to defend DoS attacks. Thus, this mechanism also realized the proposed research objective.

The last specific research objective is to validate and evaluate the performance of the proposed mechanisms in order to check their efficiency and robustness. In this chapter, we have already given an analysis of all our designed mechanisms from the aspects of techniques, attacks, security and performance (see Section 7.1, 7.2, 7.3 and 7.4). Through the analysis, we can see that our designed mechanisms are efficient, secure and robust.

The main research objective is to design a set of lightweight and efficient security mechanisms dedicated to next generation mobile networks. As we discussed above, all our designed mechanisms use only lightweight techniques on mobile devices. The performance discussion also proved that they were efficient and could provide strong

security compared to related works. All these mechanisms are dedicated to next-generation mobile networks. As we realized all the specific research objectives, we realized naturally the main research objective of this entire thesis: design a set of lightweight and efficient security mechanisms dedicated to next generation mobile networks.

CHAPTER 8

CONCLUSIONS

This thesis first gave an introduction on next-generation mobile networks and pointed out the necessity of design appropriate security mechanisms in Chapter 1. It also declared that designed mechanisms should accommodate the constraints particular in mobile networks. Open problems and research challenges were also discussed in this chapter. We are particularly interested in three issues: mobile service access, improving UMTS AKA and against DoS attacks in UMTS. For each issue, a novel mechanism is designed while accommodating constraints in mobile networks.

8.1 Summary of Contributions

This thesis has four main contributions to the state of art in research in next-generation mobile networks:

1. A novel ticket-based mobile service access mechanism;
2. An Enhancement of UMTS AKA through vector combination;
3. Mobility-Oriented AKA in UMTS;
4. A client puzzle against DoS attacks in UMTS.

The mobile service access through reusable tickets is lightweight for the mobile device and mutually authenticable for both the mobile device and the service provider. Hence, this mechanism satisfies the constraints and is appropriate to mobile networks. The two enhancements of UMTS AKA are based on the original version of UMTS

AKA which is designed specifically for mobile networks. The two enhancements improved the efficiency from the traffic point of view based on vector combination and mobility pattern. There is no expensive computation introduced in both mechanisms, thus they satisfy also the constraints and are appropriate to UMTS. As for the designed client puzzle, it is based on quasi partial collisions in a hash function. It is compact, concise and fine-grained, so it is also appropriate to mobile networks.

Therefore, our contributions realized the objectives defined in Chapter 1. Here, we summarize each contribution as follows.

8.1.1 Ticket-based Mobile Service Access

In current literature, all the proposed mobile service access mechanisms have some common disadvantages. First, the mobile user has to run expensive public key cryptographic operations such as Diffie-Hellmann and RSA. Second, every ticket can only be used for only once for service access. Consequently, a user has to acquire the quantity of tickets as many as the times of service requests. Some mechanisms considered the payment and billing issue. However, the micro-payment re-initialization issue was also expensive.

In this thesis, our contribution is a lightweight secure and accountable access to mobile services through reusable tickets. There are several advantages in our mechanism compared to related work. First, the ticket is reusable. This is realized through a hash chain with the ticket and maintained between the mobile user and the service provider. Thus, a ticket can be used many times for authentication until the hash chain runs out. The hash chain can also prevent replay attacks. All other mechanisms do not allow a ticket to be reused in order to prevent replay attacks.

Second, our mechanism is lightweight for mobile devices (only hash function and

symmetric encryption/decryption operations required). While in other mechanisms, public key cryptography is used at mobile devices in order to provide non-repudiation property; the mobile devices have to run expensive public key operations.

Third, the charging and billing is carefully designed through a staircase increment value function. It is practical and might avoid payment chain reinitialization. In some mechanisms, this issue remains untouched; in other mechanisms, payment chain reinitialization involves public key cryptography and is not suitable to mobile devices.

8.1.2 Enhancements of UMTS AKA

The current UMTS AKA has been criticized for its adoption of a sequence number between the mobile station and the HE/HLR. Consequently, it is vulnerable to redirection attacks and active attacks in corrupted networks. In addition, the HE/HLR becomes the bottleneck of the authentication vector generation.

In this thesis, we have two contributions to improve the UMTS AKA. The first contribution is called VC-AKA which is based on vector combination. It has two major advantages compared to UMTS AKA. First, it abandons completely the synchronization of sequence numbers which has been widely criticized in literature. Second, it liberates the HE/HLR from the traffic bottleneck through vector combination. There exist also many works to improve UMTS AKA. However, all of them did not decrease the traffic from the HE/HLR fundamentally. Due to numerous subscribers registered at the HE/HLR, VC-AKA does really improve the efficiency of the entire networks. Third, VC-AKA provides enhanced security and can defend efficiently malicious threats and attacks.

The second contribution is to improve the UMTS AKA from the mobility point of view. It was based on one observation: a single mobile user usually has a fairly

regular routine and could have a *home VLR* in which occur his most activities. Such mobile users have a high proportion of all mobile users. As a result, the home VLR can establish a trust relationship with the user's home network and authenticate the mobile user directly. Thus, we also liberate the HE/HLR from the traffic bottleneck.

8.1.3 Against DoS Attacks in UMTS

Our last contribution in this thesis is the fine-grained difficulty puzzles to defend against DoS attacks. Currently, the client puzzles proposed in literature have a gap between two neighbor difficulty levels. We designed a fine-grained difficulty puzzle. The client puzzle is concise and suitable to mobile devices. Through tunning the quasi bits and the difficulty level, we can add more difficulty levels between the gap. As a result, the system resource can be utilized as much as possible to support all user-demanded services while reducing the chance of DoS attacks.

8.2 Limitations

We claimed that our proposed ticket-based mobile service access mechanism was more efficient and lightweight compared to related works. The techniques used are hash chains and symmetric cryptography. A CA is required to issue and manage public/private keys for the ticket server and each VASP. Also, it is responsible to generate the user's long term secret key. This may bring attacks to the CA. If ever an adversary obtains the secret information from the CA, the entire system may become vulnerable. From the management point of view, the CA needs to be completely trusted by other principals. This imposes an implicit constraint while deploying the system. In practice, sometimes it is difficult to maintain the trust relationship between principals. It brings more management tasks. Also, the issue and revoke of certificates involves a set of updates in the system's databases. A possible solution

could be merging the CA into the ticket server. Each user registers to the ticket server directly. In this case, the certificate issued to the user might become a certificate issued by the ticket server. Even, we can abandon the certificate since the ticket server and the user keep a long-term secret key. Thus, both the user and the ticket server are responsible to maintain the long-term secret keys. This might also make the ticket server a target of DoS attacks. In addition, the ticket acquisition and the service provision protocols need to redesigned. As for the payment and billing, the staircase incremental value function is actually related to the user's service behavior. It is not trivial to determine the staircase for each user. Experimental results would be helpful.

In VC-AKA, although we eliminate the synchronization of sequence numbers between the MS and the HE/HLR, the MS and the VLR/SGSN need to maintain used combinations. The first procedure in VC-AKA needs to execute more tasks to establish the vectors in both the VLR/SGSN and the MS. Extra space is also required to store these vectors. If the MS roams from a VLR/SGSN to a new VLR/SGSN, the old VLR/SGSN should transfer authentication vectors and unused combinations to the new VLR/SGSN. Modification of VC-AKA is necessary to meet this scenario. As UMTS AKA, we assume that the channel between two VLR/SGSNs is secure. However, this assumption may not be appropriate for future heterogeneous networks. This also requires modification of the entire AKA.

As for MO-AKA, the mobility pattern establishment is critical to the whole system performance. In practice, this requires long-term users data tracking and analysis. Also, it is not trivial to determine the home VLR of a user. It may exists a few possible home VLRSs. According to the MO-AKA, we currently assume only one home VLR for each user. In this case, the system can just pick one as the home VLR. Although most customers have possible home VLRs, others may have no one simply because they have no clear mobility pattern. In this case, just selecting a VLR as

the home VLR may not be a wise solution. The system can just ignore this case and treat it with traditional UMTS AKA.

8.3 Future Work

In the ticket-based mobile service access mechanism, we have just discussed the possibility to merge the CA into the ticket server. We will eliminate the certificate as well as the CA. The user obtains the long-term secret key from the ticket server. For every ticket request, the user presents a nonce encrypted with his long-term secret key as a challenge to the ticket server; the latter decrypts the nonce and return a response together with a newly generated ticket. We will still keep the hash chain as it is essential for non-repudiation purpose in our mechanisms. We will also do some experiments to obtain user's service behaviors in order to determine a reasonable staircase incremental value function.

As for VC-AKA, we will consider the case that the channel between two VLRs are not secure. As VLRs are usually powerful enough to execute computationally expensive operations. We will adopt public key cryptography for each VLR in order to encrypt the channel. In this case, the VLR does not need to store the public keys of the users, it is enough to store only the certification and encryption key-pairs of the HLR and the VLR. Moreover, the IMSI and the public key of the user can be stored on the USIM in an encrypted form. This information is encrypted by the public encryption key of the HLR therefore only an authenticated VLR is able to learn the identity of the user. While in UMTS AKA, the IMSI is sent in the plain on the air interface thus an attacker can obtain the identity of the user by personalizing the VLR or simply by intercepting the message. Currently, there exists some approaches to enhance UMTS AKA with PKI. We will consult these approaches and propose a more complex version of enhanced AKA.

This issue also applies to MO-AKA. The implicit secure channel between the home VLR and the HLR, or different VLRs can be enhanced by public key cryptographic techniques. In addition, we care one more thing in MO-AKA, the user mobility pattern. To obtain the mobility pattern is not a trivial work. We will design a data model and a mobility profile for each user. In the data model, a location query is executed every intervals during a day or a week. The locations will be classified in order to build a rough location areas. These location areas, together with the VLR locations, will be the references to determine the possible home VLRs. We believe that mobility pattern will have high impact on the performance of MO-AKA.

BIBLIOGRAPHY

[1] G. Schäfer, "Research challenges in security for next generation mobile networks," in *PAMPAS '02 - Workshop on Requirements for Mobile Privacy & Security*, Royal Holloway, Sep. 2002.

[2] K. Martin, B. Preneel, C. Mitchell, H. Hitz, A. Poliakova, and P. Howard, "Secure billing for mobile information services in UMTS," in *Proc. 5th Int. Conf. Intelligence Services Networks Technology for Ubiquitous Telecom Services*, Antwerp, Belgium, May 1998, pp. 535–548.

[3] T. P. Pedersen, "Electronic payments of small amounts," in *Proc. the 4th Security Protocols International Workshop (Security Protocols), Lecture Notes in Computer Science*, vol. 1189, Cambridge, UK., 1996, pp. 59–68.

[4] B. Lee, T. Kim, and S. Kan, "Ticket based authentication and payment protocol for mobile telecommunications systems," in *Proc. Pacific Rim International Symposium, Dependable Computing*, Dec. 17–19, 2001, pp. 218–221.

[5] H. Wang, J. Cao, and Y. Zhang, "Ticket-based service access scheme for mobile users," *Australian Computer Science Communications*, vol. 24, no. 1, pp. 285–292, Jan./Feb. 2002.

[6] H. Wang, Y. Zhang, J. Cao, and V. Varadharajan, "Achieving secure and flexible m-services through tickets," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 33, no. 6, pp. 697–708, 2003.

[7] L. Buttyán and J. Hubaux, "Accountable anonymous service usage in mobile communication systems," in *Proc. 18th Symp. Reliable Distributed Systems*, Lausanne, Switzerland, Oct. 1999, pp. 384–389.

[8] *3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; 3G Security; Security architecture (Release 7)*, 3GPP Std. TS 33.102 V7.0.0, Dec. 2005.

[9] M. Zhang and Y. Fang, "Security analysis and enhancements of 3GPP authentication and key agreement protocol," *IEEE transactions on wireless communications*, vol. 4, no. 2, pp. 734–742, Mar. 2005.

[10] L. Harn and W. Hsin, "On the security of wireless network access with enhancements," in *Proceedings of the 2003 ACM workshop on Wireless security*, San Diego, USA, Sep. 19 2003, pp. 88–95.

[11] C. Huang and J. Li, "Authentication and key agreement protocol for UMTS with low bandwidth consumption," in *19th International Conference on Advanced Information Networking and Applications (AINA 2005)*. Taipei, Taiwan: IEEE Computer Society, Mar. 28–30, 2005, pp. 392–397.

[12] L. Lamport, "Password authentication with insecure communication," *Communications ACM*, vol. 24, no. 11, pp. 770–772, 1981.

[13] H. Krawczyk, M. Bellare, and R. Canetti, "Keyed-hashing for message authentication," *Internet Engineering Task Force, Request for Comments (RFC) 2104*, Feb. 1997.

[14] W. Diffie, P. C. van Oorschot, and M. J. Wiener, "Authentication and authenticated key exchanges," *Designs, Codes and Cryptography*, vol. 2, no. 2, pp. 107–125, 1992.

[15] T. Aura, P. Nikander, and J. Leiwo, "DoS-resistant authentication with client puzzles," in *8th International Workshop on Security Protocols*. Springer-Verlag, 2000, pp. 170–181.

[16] W. Feng, E. Kaiser, W. Feng, and A. Luu, "The design and implementation of network puzzles," in *Proc. INFOCOM'05, 24th Annual Joint Conference of the IEEE Computer and Communications Societies*, 2005.

[17] A. Juels and J. Brainard, "Client puzzles: A cryptographic defense against connection depletion," in *Network and Distributed System Security*, San Diego, 1999, pp. 151–165.

[18] X. Wang and M. Reiter, "Defending against denial-of-service attacks with puzzle auctions," in *Proc. IEEE Security and Privacy*, 2003, pp. 78–92.

[19] R. Rivest, A. Shamir, and D. Wagner, "Time-lock puzzles and timed-release crypto," in *MIT/LCS/TR-684*, 1996.

[20] B. Waters, A. Juels, J. Halderman, and E. Felten, "New client puzzle outsourcing techniques for DoS resistance," in *Proc. Computer and Communications Security*, 2004, pp. 246–256.

[21] B. C. Neuman and T. Ts'o, "Kerberos: An authentication service for computer networks," *IEEE Communications*, vol. 32, no. 9, pp. 33–38, Sep. 1994.

[22] M. Rahnema, "Overview of the GSM system and protocol architecture," *IEEE Communications Magazine*, vol. 31, pp. 92–100, 1993.

[23] C. H. Lee, M. S. Hwang, and W. P. Yang, "Enhanced privacy and authentication for the global system for mobile communications," *Wireless networks*, vol. 5, pp. 231–243, 1999.

[24] B. Patel and J. Crowcroft, "Ticket based service access for the mobile user," in *Proceedings of the 3rd annual ACM/IEEE international conference on Mobile computing and networking*, Budapest, Hungary, Sep. 26–30, 1997, pp. 223–233.

[25] H. Luo, J. Kong, P. Zerfos, S. Lu, and L. Zhang, "URSA: ubiquitous and robust access control for mobile ad hoc networks," *IEEE/ACM Transactions on Networking*, vol. 12, no. 6, pp. 1049–1063, Dec. 2004.

[26] K. Chang, T. Lee, B. Chun, and T. Kim, "Ticket-based secure delegation service supporting multiple domain models," in *Proceedings. 2001 Pacific Rim International Symposium on Dependable Computing*, Dec. 17–19, 2001, pp. 289–292.

[27] W. Lootah, W. Enck, and P. McDaniel, "TARP: Ticket-based address resolution protocol," in *21st Annual Computer Security Applications Conference*, Dec. 05–09, 2005, pp. 106–116.

[28] T. Komori and T. Saito, "A secure wireless LAN system retaining privacy," in *18th International Conference on Advanced Information Networking and Applications*, vol. 2, 2004, pp. 370–375.

[29] M. Burrows, M. Abadi, and R. Needham, "A logic of authentication," *ACM Transactions on Computer Systems*, vol. 8, no. 1, pp. 18–36, Feb. 1990.

[30] H. Lamport, "Password authentication with insecure communication," *Communications of ACM*, vol. 24, no. 11, pp. 770–772, Nov. 1981.

[31] N. Haller, "The S/KEY one-time password system," in *Proc. ISOC Symposium on Network and Distributed System Security*, Feb. 1994, pp. 151–157.

[32] R. Anderson, C. Manifavas, and C. Southerland, "Netcard - a practical electronic cash system," in *Proc. International Workshop on security Protocols*, Cambridge, UK., Apr. 10–12, 1996, pp. 49–57.

[33] G. Horn and B. Preneel, "Authentication and payment in future mobile systems," in *Proc. 5th European Symposium on Research in Computer Security*, vol. 1485, 1998, pp. 277–293.

[34] R. Rivest and A. Shamir, "Payword and micromint: two simple micropayment schemes," in *Proc. the 4th Security Protocols International Workshop (Security Protocols), Lecture Notes in Computer Science*, vol. 1189, Cambridge, UK., 1996, pp. 69–87.

[35] J. Zhou and K. Lam, "Undeniable billing in mobile communication," in *Proc. 4th ACM/IEEE International Conference on Mobile Computing and Networking*, Dallas, Texas, Oct. 1998, pp. 284–290.

[36] R. Hauser, M. Steiner, and M. Waidner, "Micro-payments based on iKP," in *14th Worldwide Congress on Computer and Communications Security Protection*, Paris, 1996, pp. 67–82.

[37] N. Asokan, G. Tsudik, and M. Waidners, "Server-supported signatures," *Journal of Computer Security*, Nov. 1997.

[38] L. Harn and H. Lin, "A non-repudiation metering scheme," *Communications Letters (IEEE)*, vol. 5, no. 12, pp. 486–487, Dec. 2001.

[39] N. Perrig, "The BiBa one-time signature and broadcast authentication protocol," in *Proc. Annual Conference on Computer and Communications Security (ACM)*, 2001, pp. 28–37.

[40] X. Ding, D. Mazzocchi, and G. Tsudik, "Experimenting with server-aided signatures," in *Network and Distributed Systems Security Symposium (NDSS '02)*, 2002.

[41] A. Perrig, R. Canetti, D. Song, and D. Tygarl, "Efficient and secure source authentication for multicast," in *Proc. Network and Distributed System Security Symposium NDSS 2001*, Feb. 2001.

[42] W. Aiello, S. Lodha, and R. Ostrovsky, "Fast digital identity revocation," in *Proc. Advances in Cryptography-Crypto '98, Lecture Notes in Computer Science*, vol. 1462, Berlin, Germany, 1998, pp. 137–152.

[43] D. Wong and A. Chan, "Mutual authentication and key exchange for low power wireless communications," in *Proc. IEEE MILCOM*, vol. 1, 2001, pp. 39–43.

[44] K. Bicakci and N. Baykal, "Infinite length hash chains and their applications," in *Proc. 11th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative 7 Enterprises (WETICE'02)*, Pittsburgh, USA, Jun. 2002, pp. 57–61.

[45] V. Goyal. How to re-initialize a hash chain. [Online]. Available: http://eprint.iacr.org/2004/097.pdf

[46] Y. Zhao and D. Li. An improved elegant method to re-initialize hash chains. [Online]. Available: http://eprint.iacr.org/2005/011.pdf

[47] C. Boyd and W. Mao, "On a limitation of BAN logic," in *In Advances in Cryptology: Eurocrypt '93*. Springer-Verlag, 1993, pp. 240–247.

[48] C. Neuman, T. Yu, S. Hartman, and K. Raeburn. The Kerberos network authentication service (v5). [Online]. Available: http://www.ietf.org/rfc/rfc4120.txt

[49] R. Needham and M. Schroeder, "Using encryption for authentication in large networks of computers," *Communications of the ACM*, vol. 21, no. 12, pp. 993–999, 1978.

[50] L. Zhu, P. Leach, and K. Jaganathan. (2006, Jul.) Anonymity support for Kerberos. [Online]. Available: http://www.ietf.org/internet-drafts/draft-ietf-krb-wg-anon-01.txt

[51] C. Neuman, B. Tung, J. Wray, and J. Trostle. Public key cryptography for initial authentication in Kerberos. [Online]. Available: ftp://ietf.org/internetdrafts/draft-ietf-cat-kerberos-pk-init-02.txt

[52] M. Sirbu and J. C. Chuang, "Distributed authentication in Kerberos using public key cryptography," in *Symposium on Network and Distributed System Security, IEEE Computer Society Press.*, San Diego, California, 1997.

[53] L. Zhu and B. Tung. Public key cryptography for initial authentication in Kerberos (PKINIT). [Online]. Available: http://www.ietf.org/rfc/rfc4556.txt

[54] I. Downnard, "Public-key cryptography extensions into Kerberos," *IEEE Potentials*, vol. 21, no. 5, pp. 30–34, Dec. 2002.

[55] L. Salgarelli, M. Buddhikot, J. Garay, S. Patel, and S. Miller, "Efficient authentication and key distribution in wireless IP networks," *IEEE Wireless Communications*, vol. 10, no. 6, pp. 52–61, Dec. 2003.

[56] G. M. Koien and T. Haslestad, "Security aspects of 3G-WLAN interworking," *IEEE Communications Magazine*, vol. 41, no. 11, pp. 82–88, Nov. 2003.

[57] J. Al-Muhtadi, D. Mickunas, and R. Campbell, "A lightweight reconfigurable security mechanism for 3G/4G mobile devices," *IEEE Wireless Communications*, vol. 9, no. 2, pp. 60–65, Apr. 2002.

[58] Y.-B. Lin, M.-F. Chang, M.-T. Hsu, and L.-Y. Wu, "One-pass GPRS and IMS authentication procedure for UMTS," *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 6, pp. 1233–1239, Jun. 2005.

[59] H. Chen, M. Zivkovic, and D.-J. Plas, "Transparent end-user authentication across heterogeneous wireless networks," in *Vehicular Technology Conference, VTC 2003-Fall*, vol. 3, Oct. 6–9, 2003, pp. 2088–2092.

[60] Y.-B. Lin, M.-F. Chen, and H.-H. Rao, "Potential fraudulent usage in mobile telecommunications networks," *IEEE IEEE Transactions on Mobile Computing*, vol. 1, no. 2, pp. 123–131, Apr.-Jun. 2002.

[61] F. Fitzek, M. Munari, V. Pastesini, S. Rossi, and L. Badia, "Security and authentication concepts for UMTS/WLAN convergence," in *Vehicular Technology Conference, VTC 2003-Fall*, vol. 4, Oct. 6–9, 2003, pp. 2343–2347.

[62] S.-H. Lim, H. S. Roh, D.-I. Kim, and S. ho Lee, "Authentication architecture for interworking between universal mobile telecommunications systems (UMTS) and high-speed portable internet (HPI) system," in *Vehicular Technology Conference, VTC 2005-Fall*, vol. 2, Sep. 25–28, 2005, pp. 774–778.

[63] K. Boman, G. Horn, P. Howard, and V. Niemi, "UMTS security," *Electronics & Communication Engineering Journal*, pp. 191–204, Oct. 2002.

[64] G. Rose and G. Koien, "Access security in CDMA2000, including a comparison with UMTS access security," *IEEE Wireless Communications*, vol. 11, no. 1, pp. 19–25, Feb. 2004.

[65] G. M. Koien, "An introduction to access security in UMTS," *IEEE Wireless Communications*, vol. 1, no. 1, pp. 8–18, Feb. 2004.

[66] Y.-B. Lin and Y.-K. Chen, "Reducing authentication signaling traffic in third-generation mobile network," *IEEE Transactions on Wireless Communications*, vol. 2, no. 3, pp. 493–501, May 2003.

[67] J. Al-Saraireh, S. Yousef, and M. A. Nabhan, "Analysis and enhancement of authentication algorithms in mobile networks," *World Wireless Congress*, vol. 2006, pp. 225 – 230, 2006.

[68] E. Halepovic and C. Williamson, "Characterizing and modeling user mobility in a cellular data network," in *Proceedings of 2nd ACM Workshop on Performance*

Evaluation of Wireless Ad Hoc, Sensor, and Ubiquitous Networks (PE-WASUN 2005), Oct. 10–13, 2005, pp. 71–78.

[69] *3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; 3G Security; Integration guidelines (Release 4)*, 3GPP Std. TS 33.103 V4.2.0, Sep. 2001.

[70] N. Eagle and A. Pentland, "Reality mining: Sensing complex social systems," *Journal of Personal and Ubiquitous Computing*, Jun. 2005.

[71] A. Quintero, "A user pattern learning strategy for managing users' mobility in UMTS networks," *IEEE Transactions on Mobile Computing*, vol. 4, no. 6, pp. 552–566, Nov.-Dec. 2005.

[72] D. Tang and M. Baker, "Analysis of a metropolitan-area wireless network," *Wireless Networks*, pp. 107–120, 2002.

[73] K. D. Simler and S. E. Czerwinski, "Analysis of wide area user mobility patterns," in *Sixth IEEE Workshop on Mobile Computing Systems and Applications*, Dec. 2–3, 2004, pp. 30–40.

[74] M. Naghshineh and M. Schwartz, "Distributed call admission control in mobile/wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 14, pp. 711–717, 1996.

[75] X. Shen, J. Mark, and J. Ye, "User mobility profile prediction: an adaptive fuzzy inference approach," *Wireless Networks*, vol. 6, pp. 363–374, 2000.

[76] J. Markoulidakis, G. Lyberopoulos, D. Tsirkas, and E. Sykas, "Mobility modeling in third-generation mobile telecommunications systems," *IEEE Personal Communications*, vol. 4, no. 4, pp. 41–56, Aug. 1997.

[77] E. Cayirci and I. F. Akyildiz, "User mobility pattern scheme for location update and paging in wireless systems," *IEEE Transactions on Mobile Computing*, vol. 1, no. 3, pp. 236–247, Jul.-Sep. 2002.

[78] G. Pollini and I. Chih-Lin, "A profile-based location strategy and its performance," *IEEE Journal on Selected Areas in Communications*, vol. 15, no. 8, pp. 1415–1424, 1997.

[79] S. Sen, A. Bhattacharya, and S. Das, "A selective location update strategy for PCS users," *ACM Journal of Wireless Networks*, vol. 5, no. 5, pp. 313–326, 1999.

[80] V. Wong and V. Leung, "An adaptive distance-based location update algorithm for next-generation PCS networks," *IEEE Journal on Selected Areas in Communications*, vol. 19, no. 10, pp. 1942–1952, 2001.

[81] P. Inc. Treo smartphones. [Online]. Available: http://www.palm.com/us/

[82] S. Ltd. Symbian smartphones for the enterprise. [Online]. Available: http://www.symbian.com/symbianos/index.html

[83] M. Corp. Windows mobile-based smartphones. [Online]. Available: http://www.microsoft.com/windowsmobile/smartphone

[84] C. Meadows, "A cost-based framework for analysis of denial of service in networks," *Journal of Computer Security*, vol. 9, no. 1, pp. 143–164, 2001.

[85] V. Gligor, "Guaranteeing access in spite of service-flooding attacks," in *Proc. Security Protocols Workshop*, 2003.

[86] C. Fung and M. Lee, "A denial-of-service resistant public-key authentication and key establishment protocol," in *21st IEEE International Performance, Computing, and Communications Conference*, Apr. 3–5, 2002, pp. 171–178.

[87] A. Mahimkar and V. Shmatikov, "Game-based analysis of denial-of-service prevention protocols," in *18th IEEE Workshop Computer Security Foundations, CSFW-18 2005.*, Jun. 20–22, 2005, pp. 287–301.